George Bebis Richard Boyle
Bahram Parvin Darko Koracin
Paolo Remagnino Ara Nefian
Gopi Meenakshisundaram Valerio Pascucci
Jiri Zara Jose Molineros
Holger Theisel Thomas Malzbender (Eds.)

LNCS 4291

Advances in Visual Computing

Second International Symposium, ISVC 2006
Lake Tahoe, NV, USA, November 2006
Proceedings, Part I

1 Part I

Springer

George Bebis Richard Boyle
Bahram Parvin Darko Koracin
Paolo Remagnino Ara Nefian
Gopi Meenakshisundaram Valerio Pascucci
Jiri Zara Jose Molineros
Holger Theisel Thomas Malzbender (Eds.)

Advances in Visual Computing

Second International Symposium, ISVC 2006
Lake Tahoe, NV, USA, November 6-8, 2006
Proceedings, Part I

Springer

Volume Editors

George Bebis
University of Nevada, Reno, USA, E-mail: bebis@cse.unr.edu

Richard Boyle
NASA Ames Research Center, CA, USA, E-mail: Richard.Boyle@nasa.gov

Bahram Parvin
Lawrence Berkeley National Laboratory, CA, USA, E-mail: parvin@hpcrd.lbl.gov

Darko Koracin
Desert Research Institute, Reno, NV, USA, E-mail: darko@dri.edu

Paolo Remagnino
DIRC, Kingston University, UK, E-mail: P.Remagnino@kingston.ac.uk

Ara Nefian
Intel, Santa Clara, CA, USA, E-mail: ara.nefian@intel.com

Gopi Meenakshisundaram
University of California at Irvine, CA, USA, E-mail: gopi@ics.uci.edu

Valerio Pascucci
Lawerence Livermore National Laboratory, USA, E-mail: pascucci1@llnl.gov

Jiri Zara
Czech Technical University in Prague, E-mail: zara@fel.cvut.cz

Jose Molineros
Rockwell Scientific, CA, USA, E-mail: jmolineros@rwsc.com

Holger Theisel
Max-Planck Institut für Informatik, Germany, E-mail: theisel@mpi-sb.mpg.de

Thomas Malzbender
Hewlett Packard Labs, Palo Alto, CA, USA, E-mail: malzbend@hpl.hp.com

Library of Congress Control Number: 2006935880

CR Subject Classification (1998): I.4, I.5, I.2.10, I.3.5, I.2.6, F.2.2

LNCS Sublibrary: SL 6 – Image Processing, Computer Vision, Pattern Recognition, and Graphics

ISSN 0302-9743
ISBN-10 3-540-48628-3 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-48628-2 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11919476 06/3142 5 4 3 2 1 0

Preface

It is with great pleasure that we welcome you all to the proceedings of the 2nd International Symposium on Visual Computing (ISVC 2006) held in Lake Tahoe. Following a successful meeting last year, we witnessed a much stronger and more productive event this year. ISVC offers a common umbrella for the four main areas of visual computing including vision, graphics, visualization, and virtual reality. Its goal is to provide a forum for researchers, scientists, engineers and practitioners throughout the world to present their latest research findings, ideas, developments and applications in the broader area of visual computing.

This year, the program consisted of 13 oral sessions, one poster session, ten special tracks, and six keynote presentations. The response to the call for papers was very strong. We received more than twice the papers received last year. Specifically, we received over 280 submissions for the main symposium from which we accepted 65 papers for oral presentation (23% acceptance) and 56 papers for poster presentation (20% acceptance). Special track papers were solicited separately through the Organizing and Program Committees of each track. A total of 57 papers were accepted for presentation in the special tracks.

All papers were reviewed with an emphasis on potential to contribute to the state of the art in the field. Selection criteria included accuracy and originality of ideas, clarity and significance of results, and presentation quality. The review process was quite rigorous, involving two to three independent blind reviews followed by several days of discussion. During the discussion period we tried to correct anomalies and errors that might have existed in the initial reviews. Despite our efforts, we recognize that some papers worthy of inclusion may have not been included in the program. We offer our sincere apologies to authors whose contributions might have been overlooked.

We wish to thank everybody who submitted their work to ISVC 2006 for review. It was because of their contributions that we succeeded in having a technical program of high scientific quality. In particular, we would like to thank the ISVC 2006 area Chairs, the organizing institutions (UNR, DRI, LBNL, and NASA Ames), our industrial sponsors (Intel, DigitalPersona, Equinox, Ford, Siemens, Hewlett Packard, NVIDIA, MERL, UtopiaCompression), the international Program Committee, the special track organizers and their Program Committees, the keynote speakers, the reviewers, and especially the authors that contributed their work to the symposium. In particular, we would like to thank Siemens who kindly offered the best paper award this year.

We sincerely hope that the proceedings of ISVC 2006 will offer opportunities for professional growth.

August 2006 ISVC 2006 Steering Committee and Area Chairs

Organization

ISVC 2006 Steering Committee

George Bebis, University of Nevada, Reno, USA
Richard Boyle, NASA Ames Research Center, USA
Bahram Parvin, Lawrence Berkeley National Laboratory, USA
Darko Koracin, Desert Research Institute, USA

ISVC 2006 Area Chairs

Computer Vision
Ara Nefian, Intel, USA
Paolo Remagnino, DIRC, Kingston University London, UK

Computer Graphics
Gopi Meenakshisundaram, University of California-Irvine, USA
Valerio Pascucci, Lawrence Livermore National Laboratory, USA

Virtual Reality
Jiri Zara, Czech Technical University in Prague, Czech Republic
Jose Molineros, Rockwell Scientific, USA

Visualization
Holger Theisel, Max-Planck-Institut für Informatik, Germany
Tom Malzbender, Hewlett Packard Labs, USA

Publicity/Website
Ali Erol, eTreppid Technologies, USA

Local Arrangements
Kostas Veropoulos, Desert Research Institute, USA

Publications
Junxian Wang, UtopiaCompression, USA

ISVC 2006 Keynote Speakers

Carolina Cruz-Neira, University of Louisiana at Lafayette, USA
Eli Peli The Schepens, Harvard Medical School, USA
Daniel DeMenthon, National Science Foundation, USA
Chris Johnson, University of Utah, USA

Dr. Karel Zuiderveld, Vital Images, USA
Mark Nixon, University of Southampton, UK

ISVC 2006 International Program Committee

(Area 1) Computer Vision

J. K. Aggarwal , University of Texas, Austin, USA
Ioannis Pavlidis, University of Houston, USA
Mubarak Shah, University of Central Florida, USA
George Bebis, University of Nevada, Reno, USA
Hammoud, Delphi Corporation, USA
Salil Prabhakar, DigitalPersona Inc., USA
GianLuca Foresti, University of Udine, Italy
Andrea Salgian, The College of New Jersey, USA
Carlo Regazzoni, University of Genoa, Italy
Tieniu Tan, Chinese Academy of Sciences, China
Mircea Nicolescu, University of Nevada, Reno, USA
Stefanos Kollias, National Technical University of Athens, Greece
Bogdan Georgescu, Siemens, USA
James Davis, Ohio State University, USA
Davide Maltoni, University of Bologna, Italy
Alessandro Verri, University of Genova, Italy
Eam Khwang Teoh, Nanyang Technological University, Singapore
Sergio Velastin, Kingston University London, UK
Nikos Paragios, Ecole Centrale de Paris, France
Nikolaos Bourbakis, ITRI Wright State University, USA
Antonis Argyros, University of Crete , Greece
Rahul Singh, San Francisco State University, USA
Zehang Sun, eTreppid Technologies, USA
Bahram Parvin, Lawrence Berkeley National Laboratory, USA
Alexei Skourikhine, Los Alamos National Lab, USA
Theodoros Katsaounis, University of Crete, Greece
Anders Heyden, Lund University, Sweden
Yoshinori Kuno, Saitama University, Japan
Gang Qian, Arizona State University, USA
Vijayan Asari, Old Dominion University, USA
Kyungnam Kim, IPIX, USA
How Lung Eng, Institute for Infocomm Research, Singapore
George Kamberov, Stevens Institute of Technology, USA
Guoliang Fan, Oklahoma State University, USA
Andrea Cavallaro, Queen Mary, University of London, UK
Larry Davis, University of Maryland, USA
Yunqian Ma, Honyewell Labs, USA
Gerald Schaefer, Nottingham Trent University, UK
Goh Wooi Boon, Nanyang Technological University, Singapore

Wei-Yun Yau, Institute for Infocomm Research, Singapore
Jochen Triesch, University of California-San Diego, USA
Michael Webster, University of Nevada, Reno, USA
Jeff Mulligan, NASA Ames Research Center, USA
Stefano Tubaro, DEI, Politecnico di Milano, Italy
Augusto Sarti, DEI, Politecnico di Milano, Italy
James Ferryman, Reading University, UK
Murat Kunt, EPFL, Switzerland
Justus Piater, Université de Liège, Belgium
Ioannis Pitas, Aristotle University of Thessaloniki, Greece
Larry Wolff, Equinox Corporation, USA
Fatih Porikli, MERL, USA
Euripides Petrakis, Technical University of Crete, Greece
Barbara Lynn O'Kane, US Army Night Vision Lab, USA
Besma Abidi, University of Tennessee, USA
Alberto Broggi, Università di Parma, Italy
Gerard Medioni, University of Southern California, USA
Peggy Agouris, University of Maine, USA
Rama Chellappa, University of Maryland, USA
Bob Fisher, University of Edinburgh, UK
Song Wang, University of South Carolina, USA
Peter Sturm, INRIA Rhône-Alpes, France
Mark Nixon, University of Southampton, UK
Ioannis Kakadiaris, University of Houston, USA
David Nister, University of Kentucky, USA
Majid Mirmehdi, Bristol University, UK
Hammadi Nait-Charif, Bournemouth University, UK
Steve Maybank, Birkbeck College, UK
Seong-Whan Lee, Korea University, Korea
Gerda Kamberova, Hofstra University, USA
Aly A. Farag, University of Louisville, USA
Dimitris Samaras, Stony Brook University, USA
Ahmed El-Gammal, University of New Jersey, USA
Christian Debrunner, Colorado School of Mines, USA
Ping Peng, Tulane University, USA
Mohammed Yeasin, University of Memphis, USA
Reinhard Klette, Auckland University, New Zeland
Kokichi Sugihara, University of Tokyo, Japan
Yunhong Wang, Chinese Academy of Sciences, China
Anders Heyden, Malmö University, Sweden
Kenneth Wong, University of Hong Kong, Hong Kong
Kenneth Tobin, Oak Ridge National Laboratory, USA
George Anagnostopoulos, Florida Institute of Technology, USA
Tanveer Syeda-Mahmood, IBM Almaden, USA
David Thirde, Reading University, UK

(Area 2) Computer Graphics

Philippe Palanque, University of Paul Sabatier, France
David Ebert, Purdue University, USA
Ik Soo Lim, University of Wales, UK
Ross Brown, Queensland University of Technology, Australia
Alexander Belyaev, Max-Planck-Institut für Informatik, Germany
Alexei Sourin, Nanyang Technological University, Singapore
Ming Wan, Boeing, USA
Irene Cheng, University of Alberta, Canada
Min-Hyung Choi, University of Colorado at Denver, USA
Jim Cremer, University of Iowa, USA
Andre Hinkenjan, Bonn-Rhein-Sieg University of Applied Sciences, Germany
Han-Wei Shen, Ohio State University, USA
Holly Rushmeier, Yale University, USA
Issei Fujishiro, Tohoku University, Japan
John C Hart, University of Illinois at Urbana-Champaign, USA
Kelly Gaither, University of Texas at Austin, USA
Leila De Floriani, University of Maryland, USA
Rachael Brady, Duke University, USA
Raghu Machiraju, Ohio State University, USA
Arik Shamir, The Interdisciplinary Center, Herzliya, Israel
Claudio Silva, University of Utah, USA
Jim Ahrens, Lawrence Livermore National Laboratory, USA
Ken Joy, University of California, Davis USA
Renato Pajarola, University of Zurich, Switzerland

(Area 3) Virtual Reality

Anders Heyden, Lund University, Sweden
Alvar Vinacua, Universitat Politècnica de Catalunya, Spain
Miguel Otaduy, ETH-Zurich, Switzerland
Fred Harris, University of Nevada, Reno, USA
Nicholas Bilalis, Technical University of Crete, Greece
Alexei Sourin, Nanyang Technological University, Singapore
Ming Wan, Boeing, USA
Irene Cheng, University of Alberta, Canada
Richard Boyle, NASA Ames Research Center, USA
Cesar Mendoza, Universidad Rey Juan Carlos, Spain
Reinhold Behringer, Leeds Metropolitan University UK
Jos Remo Ferreira Brega, UNIVEM, PPGCC, Brazil
Hans Hagen, University of Kaiserslautern, Germany
Robert van Liere, CWI, Netherlands
Min-Hyung Choi, University of Colorado at Denver, USA
Cagatay Basdogan, Koç University, Turkey
Jim Cremer, University of Iowa, USA
Joe LaViola, Brown University, USA
Simon Richir, University of Angers, France

Manohar Srikanth, Indian Institute of Science, India
Nickolas Sapidis, Aegean University, Greece
Nigel John, University of Wales Bangor, UK
Ildeberto Rodello, UNIVEM, PPGCC, Brazil
Alan Craig, NCSA University of Illinois at Urbana-Champaign, USA
George Kartsounis, Agricultural University of Athens, Greece
Andre Hinkenjan, Bonn-Rhein-Sieg University of Applied Sciences, Germany
Joerg Meyer, University of California Irvine, USA
Roberto Ranon, University of Udine, Italy
Thomas Varsamidis, University of Wales, UK
Sabine Coquillart, INRIA, France
Greg Schmidt, Naval Research Laboratory, USA
Chunrong Yuan, Fraunhofer Inst. for Applied Info Tech., Germany

(Area 4) Visualization

J. Edward Swan II, The Naval Research Laboratory, USA
James Klosowski, IBM T.J. Watson Research Center, USA
Paolo Cignoni, ISTI - CNR, Italy
Nicholas Bilalis, Technical University of Crete, Greece
Darko Koracin, Desert Research Institute, USA
Fred Harris, University of Nevada, Reno, USA
Olaf Thiele, University of Mannheim, Germany
Robert Rabin, University of Wisconsin, Madison, USA
David Ebert, Purdue University, USA
Helwig Hauser, VRVis Research Center, Austria
Robert Moorhead, Mississippi State University, USA
Klaus Mueller, SUNY Stony Brook, USA
Theresa-Marie Rhyne, North Carolina State University, USA
Mark Apperley, University of Waikato, New Zealand
Alfred Inselberg, Tel Aviv University, Israel
Nabil Adam, Rutgers University, USA
Brian Wylie, Sandia National Laboratory, USA
Alexei Sourin, Nanyang Technological University, Singapore
Mao Lin Huang, University of Technology, Australia
Anthony Maeder, CSIRO ICT Centre, Australia
Jos Roerdink, University of Groningen, Netherlands
Jose Malpica, Alcala University, Spain
Yoshitaka Masutani, The University of Tokyo Hospital, Japan
Pavel Slavik, Czech Technical University in Prague, Czech Republic
Kwan-Liu Ma, University of California-Davis, USA
Ming Wan, Boeing, USA
Irene Cheng, University of Alberta, Canada
Jack Snoeyink, University of North Carolina, USA
Heidrun Schumann, Rostock University, Germany

Ross Brown, Queensland University of Technology, Australia
Robert van Liere, CWI, Netherlands

ISVC 2006 Special Tracks

1. Intelligent Environments: Algorithms and Applications

Organizers
Paolo Remagnino, DIRC, Kingston University, UK
How-Lung Eng, IIR, Singapore
Guoliang Fan, Oklahoma State University, USA
Yunqian Ma, Honeywell Labs, USA
Monique Thonnat, INRIA, France

2. Multimodal Data Understanding and Visualization for Industrial Applications

Organizers
Fatih Porikli, MERL, USA
Andrea Cavallaro, Queen Mary, University of London, UK

Program Committee
Rama Chellapa, University of Maryland, USA
Yuri Ivanov, MERL, USA
Swarup Medasani, HRL, USA
Ron Miller, Ford Motor Company, USA
Chris Wren, MERL, USA

3. Pattern Analysis and Recognition Applications in Biometrics

Organizers
Ali Erol, University of Nevada, Reno, USA
Salil Prabhakar, DigitalPersona, USA
Mark Nixon, University of Southampton, UK
Arun Abraham Ross, West Virginia University, USA

4. Biomedical Image Analysis

Organizers
Tao Ju, Washington University, USA
Ioannis Kakadiaris, University of Houston, USA
Shi Pengcheng, Hong Kong University of Science and Technology, China
Tomas Gustavsson, Chalmers University of Technology, Sweden

5. Understanding and Imitating Nature: Analysis, Interpretation, Rendering and Inspiration of Biological Forms

Organizers
Paolo Remagnino, DIRC, Kingston University, UK
Richard Boyle, NASA Ames, USA
Paul Wilkin, The Royal Botanic Gardens, UK
Jonathan Clark, University of Surrey, UK
Sarah Barman, Kingston University, UK

6. Visual Computing and Biological Vision

Organizers
Jeff Mulligan, NASA Ames, USA
Michael Webster, University of Nevada, Reno, USA
Alice O'Toole, University of Texas at Dallas, USA

7. 4D Medical Data Modeling, Visualization and Measurement

Organizers
Irene Cheng, University of Alberta, Canada
Randy Goebel, University of Alberta, Canada
Lijun Yin, State University of New York, USA

Program Committee
Walter Bischof, University of Alberta, Canada
Pierre Boulanger, University of Alberta, Canada
Paul Major, University of Alberta, Canada
Jana Rieger, Misericordia Community Hospital, Canada
Brian Maraj, University of Alberta, Canada
Carol Boliek, University of Alberta, Canada

8. Discrete and Computational Geometry and Their Applications in Visual Computing

Organizers
Valentin Brimkov, State University of New York, USA
Reneta Barneva, State University of New York, USA

Program Committee
Eric Andres, Université de Poitiers, France
David Coeurjolly, Université Claude Bernand Lyon, France
Isabelle Debled-Rennesson, IUFM de Lorraine, France
Guillaume Damiand, Université de Poitiers, France
Christophe Fiorio, Ecole Polytechnique Universitaire de Montpellier, France
Atushi Imiya, Chyba University, Japan
Reinhard Klette, Auckland University, New Zealand

9. Soft Computing in Image Processing and Computer Vision

Organizers
Gerald Schaefer, Nottingham Trent University, UK
Muhammad Sarfraz, King Fahd University of Petroleum and Minerals, Saudi Arabia
Lars Nolle, Nottingham Trent University, UK

10. Energy Minimization Approaches in Image Processing and Computer Vision

Organizers
Jose M. Bioucas-Dias, Instituto Superior Tecnico Torre Norte, Portugal
Antonin Chambolle, CMAP Ecole Polytechnique, France
Jerome Darbon, EPITA Research and Development Laboratory, France

Additional Reviewers

Steve Callahan
Emanuele Santos
John Schreiner
Louis Bavoil
Linh Ha
Huy T. Vo
Erik Anderson
Raphael Brger
Oliver Wang
Max Louwerse
Mike Harville
Bruce Culbertson
Harlyn Baker
Alireza Tavakkoli
Leandro Loss
Gholamreza Amayeh
Kostas Veropoulos
Junxian Wang
Ali Erol

Organizing Institutions and Sponsors

Table of Contents – Part I

Table of Contents – Part II

Activity Recognition Via Classification Constrained Diffusion Maps

Yunqian Ma[1], S.B. Damelin[2], O. Masoud[3], and N. Papanikolopoulos[3]

[1] Honeywell labs, Honeywell International Inc.,
3660 Technology Drive, Minneapolis, MN 55418
yunqian.ma@honeywell.com

[2] University of Minnesota, Institute for Mathematics and its Applications
400 Lind Hall, 207 Church Hill, S.E Minneapolis, MN 55455
damelin@georgiasouthern.edu

[3] University of Minnesota, Artificial Intelligence, Vision and Robotics Lab
Department of Computer Science and Engineering, Minneapolis, MN 55455
{masoud, npapas}@cs.umn.edu

Abstract. Applying advanced video technology to understand human activity and intent is becoming increasingly important for video surveillance. In this paper, we perform automatic activity recognition by classification of spatial temporal features from video sequence. We propose to incorporate class labels information to find optimal heating time for dimensionality reduction using diffusion via random walks. We perform experiments on real data, and compare the proposed method with existing random walk diffusion map method and dual root minimal spanning tree diffusion method. Experimental results show that our proposed method is better.

1 Introduction and Background

Recognition of human actions from video streams has recently become an active area of research with numerous applications in video surveillance, which is mostly motivated by the increasing number of video cameras deployed for video surveillance and the current inability of video operators to monitor and analyze large volumes of data. For predefined activities, many rule-based or logic based methods have been proposed. For example, in [1], the authors define a series of rules, e.g. entry violation, escort, theft whereas the results of [2] use a declarative model and a logic based approach to recognize predefined activities. Unfortunately a major drawback of pre-defined activity recognition approaches is that the rules developed for one activity typically may not be applicable for other activities. Indeed, different application domains may be interested in different activities. One of the key challenges in these later systems is the ability to model the activities of interest, as well as develop a methodology that allows automatic recognition of activities. In [3,4,5], the authors show that same or similar activity video sequences are clustered close to each other and far from different activity video sequences. This paper targets an automatic activity recognition system which initially has an activity gallery that may be empty or may contain a number of initial simple activities. The system is trained by example, where input video sequences are manually labeled and the system extracts features and automatically learns the new activity.

We suppose that we are given a set of labeled video sequences as training data. The class label denotes a number of activities. Each video sequence is represented by a high dimensional feature considered isometric to a point in a high dimensional vector space. One may think that high dimension here should be an obstacle for any efficient processing of our data. Indeed, many machine learning algorithms have a computational complexity that grows exponentially with dimension. Dimensionality reduction is a way to find an isometric mapping of each video sequence into a corresponding point in Euclidean space of lower dimension where its description is considered simpler.

High dimensional spatial temporal features are associated with the nodes of a graph with a natural metric. After dimensionality reduction, a classifier (e.g. k nearest neighbor classifier) is performed on the reduced features. Using dimensionality reduction in the application of activity recognition can be found in Zhong et al. and Porikli et al. [6,7]. For example, in [6] the authors calculate the co-occurrence matrix between features, and solve for the smallest eigenvectors to find an embedding space.

In this paper, we propose a new dimensionality reduction method. The idea is to incorporate class labels information in the training data to find the optimal heating time t for dimensionality reduction using diffusion via random walks. For each heating time t, it associates a map which takes high dimensional feature to a reduced feature points. With the class labels, we perform cross validation method on the training data and then select the optimal t value which yields the smallest cross validation value. For this optimal t, we perform diffusion dimensionality reduction on the high dimensional spatial temporal feature and then use a k nearest neighbor classifier on the reduced space. We use our methods on real data, and compare the proposed method with existing random walk diffusion and dual root minimal spanning tree diffusion.

The remainder of this paper is organized as follows. In Section 2, we first describe spatial temporal features and then describe existing diffusion map methods. Section 3 describes our proposed classification constrained diffusion map method. In Section 4, we present experimental results and finally in Section 5, we present a summary.

2 Existing Diffusion Map Methods

Before we describe the existing diffusion map methods, let's briefly talk about the high dimensional Spatial temporal features used in this paper.

Davis and Bobick [10]used recursive filtering to construct feature images that represent motion: recent motion is represented as brighter than older motion. We use a similar approach described in [12]. Actions can be complex and repetitive making it difficult to capture motion details in one feature image. In this method, a weighted average at time $i \geq 1$, M_i is computed as $M_i = \alpha I_{i-1} + (1-\alpha)M_{i-1}$, where I_i is the image at time i and $0 \leq \alpha \leq 1$ is a fixed scalar. The feature image at time i, which we denote by F_i is computed as $F_i = |I_i - M_i|$. Note that it is the contrast of the gray level of the moving object which determines the magnitude of the feature image not the actual gray level value. To form a spatial

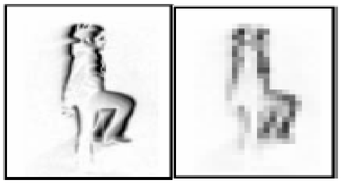


Fig. 1. An original size and a reduced size feature image from a an action of marching soldiers

temporal features, we can combine L frames together, e.g. $L = 12$ in this paper. So each spatial temporal feature is isometric to a point in $\mathbb{R}^{25\times31\times12}$, where spatial resolution can be reduced to 25x31 pixels [12], as shown in Figure 1.

Next, we describe Diffusion via Random Walks [9]. We denote X is the space of spatial temporal features in $\mathbb{R}^d$. First introduced in the context of manifold learning, eigenmap techniques [8,9,11] are methods to isometrically embedd points of X into a lower dimensional Euclidean space. Spectral methods take into account local distortion of data points in X. The diffusion map methods belong to the spectral methods.

The existing diffusion map methods of Diffusion via random walk [9] is that it constructs a graph on X where each point is considered a node and every two nodes are connected by an edge via a non negative, symmetric, positive definite kernel $w : X \times X \to \mathbb{R}$. For example, the heat kernel can be

$$w_\sigma(\mathbf{x}_i, \mathbf{x}_j) := \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2}\right), \ \mathbf{x}_i, \mathbf{x}_j \in X, \ i, j = 1, .., n \qquad (1)$$

where σ is a kernel width parameter. The parameter σ gives the rate at which the similarity between two points decays. The weight w reflects the degree of similarity or interaction between the points $\mathbf{x}_i, \mathbf{x}_j \in X$ and depends only on the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ in X. Here, $||.||$ is the Euclidean norm in $\mathbb{R}^d$.

A Markov chain is defined on X as follows. Given a node $\mathbf{x}_i \in X$, we define the degree of $\mathbf{x}_i$ by $d(\mathbf{x}_i) = \sum_{\mathbf{x}_j \in X} w_\sigma(\mathbf{x}_i, \mathbf{x}_j)$. We then form a $n \times n$ affinity matrix P with entries $p(\mathbf{x}_i, \mathbf{x}_j) = \frac{w_\sigma(\mathbf{x}_i, \mathbf{x}_j)}{d(\mathbf{x}_i)}$, $i, j = 1, ..., n$. Because $\sum_{\mathbf{x}_j \in X} p(\mathbf{x}_i, \mathbf{x}_j) = 1$, P is a transition matrix of a Markov chain on the graph of the members of X. Taking powers of P in steps $t \geq 1$, produces probabilty functions $p_t(\mathbf{x}_i, \mathbf{x}_j)$ which measure the probability of transition from $\mathbf{x}_i$ to $\mathbf{x}_j$ in t steps. Since w_σ is symmetric, P has a sequence of n eigenvalues

$$1 \geq \lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$$

and a collection of $1 \leq d_r \leq n$ right eigenvectors $\{\phi_{d_r}\}$ so that for each fixed $t \geq 1$,

$$P^t \phi_{d_r} = \lambda_{d_r}^t \phi_{d_r}.$$

Each eigenvector is a signal over the data points and the eigenvectors form a new set of coordinates on X. For any choice of t, the mapping

$$\Psi_t : \mathbf{x}_i \to \left(\lambda_1^t \phi_1(\mathbf{x}_i), ..., \lambda_{d_r}^t \phi_{d_r}(\mathbf{x}_i)\right)^T \tag{2}$$

is an isometric embedding of X into $\mathbb{R}^{d_r}$ and the function

$$\alpha(\mathbf{x}_i, \mathbf{x}_j) := ||\Psi(\mathbf{x}_i) - \Psi(\mathbf{x}_j)||,\ i, j = 1, ..., n$$

defines a metric on the graph given by the nodes of X. Here $||.||$ denotes the Euclidean norm in $\mathbb{R}^{d_r}$.

The reason that spectral clustering methods work [13] is that with sparse kernel matrices, long range affinities are accommodated through the chaining of many local interactions as opposed to standard Euclidean distance methods - e.g. correlation - that impute global influence into each pair wise affinity metric, making long range interactions wash out local interactions.

Another diffusion methods, proposed by Grikschat et al. [11], is a dual root minimal spanning tree diffusion method, we put the description in Appendix, since we also compared our proposed method with those method.

3 Proposed Method

We are given a set of labeled video sequences $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, n$ as training data and an unlabeled set of testing data $\mathbf{x}_i$, $i = 1, \ldots, m$ where each $\mathbf{x}_i \in \mathbb{R}^d$ represents a d dimensional spatial temporal feature, and $y_i \in \{1, ..., p\}$ is a class label in p activities.

Spatial temporal features $\mathbf{x}_i$, $i = 1, .., n + m$ are associated with the nodes of a graph with a natural metric given in section 2 for dimension reduction using diffusion via random walks. As described in section 2, for each fixed positive integer t, we have a map

$$t : \mathbf{z}_i^{(t)} = \mathbf{x}_i \to \left(\lambda_1^t \phi_1(\mathbf{x}_i), ..., \lambda_{d_r}^t \phi_{d_r}(\mathbf{x}_i)\right)^T, i = 1, ..., n + m \tag{3}$$

Here, for any t, we produce from $n + m$ points $\mathbf{x}_i$ a new set of $n + m$ reduced points $\mathbf{z}_i^{(t)} \in \mathbb{R}^{d_r}$ in an Euclidean space of reduced dimension d_r. The parameters t introduce weights as multiplication of eigenvalues by eienvectors.

From another point of view, we consider the matrix P^t where P is the affinity matrix defined in Section 2, which can be viewed as a transition matrix of a Markov chain on the nodes. Taking powers of P in steps t, produces probabilty functions which measure the probability of transition from $\mathbf{x}_i$ to $\mathbf{x}_j$ in t steps. (For this reason, the maps are diffusion related). In this paper, we propose to use class label information in the training data to choose an optimal t value for dimensionality reduction. The detail is as follows:

For each $t \geq 1$, we use cross validation on the training points $(\mathbf{z}_i^{(t)}, y_i), i = 1, ..., n$. Specifically, we use leave-one-out cross validation defined by

$$CV^{(t)} := \frac{1}{n} \sum_{i=1}^{n} L(\hat{f^{-i}}(\mathbf{z}_i^{(t)}), y_i) \tag{4}$$

where $\hat{f^{-i}}$ is the fitting function computed with the ith part of the data removed and L is 1 if $\hat{f^{-i}}(\mathbf{z}_i^{(t)}) = y_i$ and 0 otherwise.

We then select the optimal t value t^{opt} defined as

$$t^{\text{opt}} = \text{argmin}(CV^{(t)}) \tag{5}$$

which yields the smallest cross validation value.

We perform t^{opt}-step random walk diffusion and arrive the resulting low dimensional feature $\mathbf{z}_i^{t^{\text{opt}}}$. After that, we use k nearest neighbor classifier for the multi-class classification on the reduce space's of the test data $\mathbf{z}_i^{t^{\text{opt}}}, i = 1, ..., m$.

4 Experimental Results

In this experiment, the video sequences were recorded using a single stationary monochrome CCD camera mounted in such a way that the actions are performed parallel to the image plane. The data set consists of actions performed by $s = 29$ different people. Each person performed $p = 8$ activities, as shown in Figure 2: walk, run, skip, line-walk, hop, march, side-walk, side-skip. The location and size of the person in the image plane is assumed to be available (e.g., through tracking). Each activity sequence by each person includes a full cycle of the activity. The number of frames per sequence therefore depends on the speed of each action.

In our experiments, we used the data for eight of the 29 subjects for training (64 video sequences). This leaves a test data set of 168 video sequences performed by the remaining 21 subjects. The training instances have label. The number of selected frames was arbitrarily set to 12. So, the full dimension d of the space is 775*12 dimensions (as shown in Section 2).

For performance evaluation, we calculated prediction risk on the test data by the formula (6):

$$R_{\text{pred}} := \frac{1}{m} \sum_{i=1}^{m} L(\hat{y}_i, y_i). \tag{6}$$

Finally we compare our new method with the existing random walk diffusion and dual root minimal spanning tree diffusion.

Table 1 shows the results of prediction risk (error) using our proposed method. Table 2 and Table 3 are the results using existing Random walk diffusion+KNN and Dual rooted diffusion+KNN;

The best result of the classification error in Table 1 is 36.31%, which is better than the corresponding best result from Table 2 and Table 3 (42.86%, 57.14%

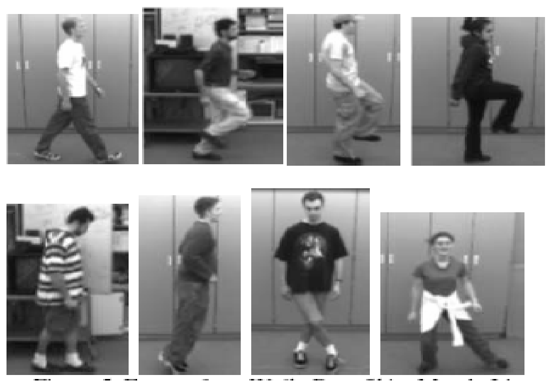


Fig. 2. Frames from walk, run, skip, march, line-walk, hop,side-walk,side-skip actions respectively

Table 1. Classification error using proposed method of cross validation to select best heating time

	d=3	d=5	d=20	d=50	d=100	d=150	d=200
$\sigma = 6$	53.57%	53.57%	51.79%	53.57%	57.74%	70.24 %	73.21%
$\sigma = 8$	57.14%	45.24%	44.05%	44.05%	46.43%	44.05%	50%
$\sigma = 10$	54.67%	56.55%	43.45%	41.07%	44.64%	47.02%	44.05%
$\sigma = 12$	50.6%	53.57%	38.69%	36.31%	39.88%	40.48%	41.67%
$\sigma = 14$	57.74%	50.6%	44.64%	47.02%	42.26%	42.26%	41.67%

Table 2. Classification error using using existing Random walk diffusion

	d=3	d=5	d=20	d=50	d=100	d=150	d=200
$\sigma = 6$	53.57%	53.57%	51.79%	56.55%	59.52%	73.21%	84.52%
$\sigma = 8$	58.93%	45.24%	50.6%	48.81%	59.52%	69.64%	87.5%
$\sigma = 10$	57.14%	56.55%	44.64%	44.64%	58.33%	69.05%	86.9%
$\sigma = 12$	62.5%	55.95%	42.86%	44.64%	60.12%	70.83%	87.5%
$\sigma = 14$	57.74%	50.6%	44.64%	47.02%	60.12%	72.62%	87.5%

Table 3. Classification error using Dual rooted diffusion, where $v = max(max(Ahop))$

	d=3	d=5	d=20	d=50	d=100	d=150	d=200
$\sigma = 1/6 * v$	64.29%	64.88%	63.69%	66.07%	65.48%	74.4%	83.33%
$\sigma = 1/8 * v$	61.9%	61.31%	63.69%	61.31%	64.88%	73.81%	88.1%
$\sigma = 1/10 * v$	61.31%	63.69%	57.14%	62.5%	66.67%	70.83%	83.93%
$\sigma = 1/12 * v$	60.71%	60.12%	60.12%	61.31%	66.07%	74.4%	82.74%
$\sigma = 1/14 * v$	59.52%	60.12%	63.1%	59.52%	64.29%	75%	82.74%

respectively). Also the results in Table 1 is stable and are considerably lower and less sensitive to the choice of σ than those of Table 2 and Table 3. The reason is that using proposed method the choice of optimal heating time t compensates for the sensitivity of σ.

We also use different k-value in the k-nearest neighbor classifier, and have the similar experimental results to the case k=3.

5 Summary

In this paper, we studied the problem of activity recognition via classification of spatial temporal feature actions in the video surveillance application. In practical application, these high level sematic learning performance also depends on the quality of low level processing, such as motion detection and motion tracking. Sometimes, there is bad quality of low level processing. For example, under very noisy video environment, shadows, occlusion, the current low level processing can not reach 100% satisfaction of the needs of activity recognition. We discuss the robust tracking algorithm in [14], so in this paper we assume the low level processing motion detection and motion tracking is well done.

In this paper, our new idea is to use class labels to find optimal heating times for dimension reduction using diffusion via random walks. We used our methods on real data, and compared our new method with the existing diffusion maps method for random walk diffusion and dual root minimal spanning tree diffusion. The results by our proposed method are shown to be considerably better as the choice of optimal heating time.

The activity we discuss in this paper is simple activity, in the future work we will perform research on complex activity, which is a combination of a series simple activities.

Acknowledgement. The authors want to thank Professor A. Hero from University of Michigan and Dr. T. Wittman from University of Minnesota for useful discussions.

References

1. V. D. Shet, D. Harwood and L. S. Davis, *VidMAP: Video Monitoring of Activity with Prolog*, IEEE International Conference on Advanced Video and Signal based Surveillance (AVSS), Como, Italy, September 2005.
2. V. Vu, F. Bremond and M. Thonnat, *Video surveillance: human behaviour representation and on-line recognition*, The Sixth International Conference on Knowledge-Based Intelligent Information and Engineering Systems, Podere d'Ombriano, Crema, Italy, September 2002.
3. Y. Ma, B. Miller, P. Buddharaju and M. Bazakos, *Activity Awareness: From Predefined Events to New Pattern Discovery*, 2006 IEEE International Conference on Computer Vision Systems, New York, USA, January 5-7, 2006.
4. R. Chellappa, A. Chowdhury and S. Zhou, *Recognition of Humans and Their Activities Using Video*, Morgan Claypool, 2005.

5. N. Jin and F. Mokhtarian, *Human Motion Recognition based on Statistical Shape Analysis*, Proc. IEEE International Conference on Advanced Video and Signal based Surveillance, pp. 4-9, 2005.
6. H. Zhong, J. Shi and M. Visontai, *Detecting Unusual Activity in Video*, IEEE Conference on Computer Vision and Pattern Recognition(CVPR), 2004.
7. F. Porikli and T. Haga, *Event detection by Eigenvector Decomposition using object and feature frame*, CVPR workshop, pp. 114-114, 2004.
8. M. Belkin, P. Niyogi, *Laplacian Eigenmaps for Dimensionality Reduction and Data Representation*, Neural Computation, June 2003; **15** (6), pp. 1373-1396.
9. R. R. Coifman, S. Lafon, *Diffusion Maps*, submitted to Applied Computational and Harmonic Analysis, 2004.
10. J. Davis and A. Bobick, *The Representation and Recognition of Action Using Temporal Templates*, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Puerto Rico, pp. 928-934, June 1997.
11. S. Grikschat, J. Costa, A. Hero and O. Michel, *Dual rooted diffusions for clustering and classification on manifolds*, 2006 IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing, Toulouse France, 2006.
12. O. Masoud, N.P. Papanikolopoulos, *A method for human action recognition*, Image and Vision Computing, **21**, no. 8, pp. 729-743, Aug. 2003.
13. A. Ng, M. Jordan and Y. Weiss, *On spectral clustering:analysis and an algorithm*, Neural Information Processing Systems, **14**, 2001.
14. Y. Ma, Q. Yu and I. Cohen, *Multiple Hypothesis Target Tracking using Merge and Split of Graphs Nodes*, 2nd International Symposium on Visual Computing, Lake Tahoe, Nevada, Nov. 6-8, 2006.

Appendix: Diffusion Via Dual Root Minimal Spanning Trees

Starting with two random walks on different points $\mathbf{x}_i$ and $\mathbf{x}_j$ in X, when will two paths generated hit each other. More precisely, given $\mathbf{x}_i \in X$, we compute a greedy minimal spanning tree and define the distance d between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ as the number of greedy iterations required so that two greedy minimal spanning trees rooted on each point $\mathbf{x}_i$ and $\mathbf{x}_j$ in X will intersect. We set σ to be $1/C \cdot \max(\max(\text{Ahop}))$ where $C > 1$ and $Ahop$ is the matrix of all pairwise distances - the hitting times between diffusions from different pairs of points. (In [11] C is taken as 10). This is an adaptive normalization in the sense that it makes the kernel decay on the order of $1/C$ of the maximum of the hitting times. An affinity matrix P is calculated with the weight w_σ with distance given by the hitting time between points x and y and the eigenvectors of P are used for a dimension reduction map.

Generating and Updating Textures for a Large-Scale Environment

Jinhui Hu, Suya You, and Ulrich Neumann

University of Southern California

Abstract. With the rapid development of sensor and modeling technologies, it becomes increasingly feasible to model a large-scale environment. However, the acquisition and updating of textures for such a large-scale environment is still a challenging task, often demanding tedious and time-consuming manual interactions. This paper presents new techniques to generate high quality textures for given rough urban building models by automatic camera calibration and pose recovery, and to continuously update these textures in real time using videos as a texture resource. A number of static textures are generated for a university campus size model, and these textures are dynamically updated using videos in real time, which demonstrate the effectiveness of our algorithms.

1 Introduction

Creating a large-scale virtual environment is important for many computer graphics and user interaction applications. With the rapid development of sensor and modeling technologies, it becomes increasingly feasible to model a large-scale environment [1]. In rendering a virtual environment, texture mapping is often employed to produce a realistic image without the tedium of modeling small-scale 3D structures and surface reflectance parameters. However, the acquisition and updating of textures for a large-scale environment is a challenging task that often demands tedious and time-consuming manual interactions.

The goal of this paper is to generate high quality textures for given urban building models, and to update these textures in real time using videos as a texture resource. The textures generated from videos enable the visualization to reflect smoothly varying of textures when the viewpoint changes and the most recent changes in the environments. The textures from both static images and videos can be mapped to urban models generated using LIDAR data [1] or aerial images [2].

1.1 Related Work

A wealth of research has been conducted on texture generation. A popular method is texture synthesis, which can be further divided into three classes. The first one is a parametric model-based technique, which uses a number of parameters to describe a variety of textures [3]. The second class is non-parametric textures, or example-based methods. These methods generate textures by directly copying pixels from input textures [4]. The third class synthesizes textures by copying whole patches from input images [5]. Texture synthesis has been shown to be a powerful tool that is widely used in many fields. However, the resulting textures are not representative of an actual scene, which limits their application for reconstructing realistic images of real scenes.

A straightforward way to generate realistic textures is to use images as the texture source [15]. Fruh and Zakhor [6] fix the image sensor with the range sensor to capture both the texture and geometry data, which has the advantage of simple calibration, but lack flexibility. Stamos and Allen [7] use a freely moving camera to capture the image data. The camera pose is computed by fitting rectangular features from dense 3D range data, which is not applicable for rough urban models. Lee et al. [2] propose a system to register ground images to urban models using vanishing points to estimate the orientation and 2D to 3D feature correspondences to estimate the translation. The under-constrained pose problem is solved by manually inferring more 3D points from registered images, which is not applicable when all images are under-constrained. Recent work [8] treats the repeated patterns in video clips as a video texture, while we are focusing on the traditional textures, i.e., pixels in images.

2 Texture Storage

Our goal is to generate a photorealistic virtual environment with textures from real images rather than generating texture patterns or synthesizing textures on-the-fly while rendering. So an efficient way to store textures for a large-scale environment is important, especially when videos are used as a texture resource. With the fact that only limited scene geometry can be visible from a viewpoint, we propose the idea of *base texture buffer*, which is a texture buffer associated with a model patch or a group of neighboring patches. The base texture buffer is first initialized as white texture, and then updated with static images (Section 3) or continuous videos (Section 4).

2.1 Base-Buffer Allocation

Given the 3D models of a large-scale environment, we need to allocate base buffers for texture storage. The goal of base-buffer allocation is to find an optimal allocation that covers all the geometry and satisfies three criteria: minimum number of base buffers, minimum number of clipped polygons, and optimal texture quality.

The number of base buffers corresponds to the number of texture units in scene rendering using graphics hardware, and is also proportional to the size of texture memory. Current graphics cards have a limited number of texture units and memory, so limiting the number of base buffers is necessary. The number of clipped polygons will increase the total number of polygons and affect the rendering speed, we also want to minimize this number. The last issue with base-buffer allocation is the texture quality problem. Texture quality depends on the video resolution, the angle of the polygon relative to the camera view direction, and the angle of the polygon relative to the base buffer. A good allocation for a base buffer is one where textures will not be compressed or stretched during the texture mapping process. This means the base buffer should be oriented parallel to most of its associated polygons.

2.2 Solution

We summarize the three criteria of base-buffer allocation using a cost function shown in Equation 1, where $f(bn)$ is a function of the number of buffers (bn), $g(pn)$ is a function of the number of clipped polygons (pn), and h is a function of the texture quality. The design of the functions $f(bn)$ and $g(pn)$ is straightforward; however, the measurement of texture quality is more complex. Our measure is the ratio of the model surface area to the texture image area, as given in Equation 2. We aim to find a buffer allocation method that minimizes the cost of Equation 1.

$$F(buffer) = f(bn) + g(pn) + h(texure\_quality)\,. \tag{1}$$

$$h(texture\_quality) = S(polygon)\,/\,S(texture)\,. \tag{2}$$

The problem of selecting optimal base buffers is analogous to the problem of selecting a minimal number of camera positions that cover a given 3D model. The latter problem has been shown to be NP-complete [9]. Since the exact allocation solution is prohibitive, we attack this problem by leveraging the characteristics of the building models. Noting that buildings usually have four major planar sides, we opt for using a box with four rectangular texture-maps for each building. To allocate the four rectangular buffers, we first automatically divide the scene model into separate buildings using connectivity. Then we find the four major directions based on building surface normals and allocate a base buffer for each direction (Figure 1). We project the vertices onto the base buffer with an orthogonal projection to compute the UV coordinate for each model vertex.

Four rectangular buffers work well for our campus model. The base buffers are automatically allocated, yielding a near-optimal solution according to the energy function in Equation 1. The campus model has 100 buildings and 40,000 polygons, covered by 400 base buffers. The texture quality is maximized for most polygons because the buffers are parallel to most of the polygon planes. Since each base buffer covers one whole side of a building, polygon clipping is avoided.

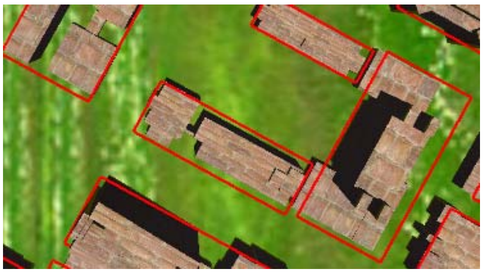


Fig. 1. Base-buffer allocation

3 Generating Static Textures

After allocating base buffers for all the models, we first update the buffers with static textures generated from real images. The key problem in generating static textures is to automatically recover the camera poses. We decompose a camera's pose into an orientation and a translation, and estimate the orientation of the camera using vanishing points extracted by automatic line clustering, and the translation using 3D to 2D corner correspondences. The camera projection matrix is modeled using Equation 3, where K is the internal parameter matrix, and RT is the camera's external parameter matrix. The focal length and principle point can be estimated given three orthogonal vanishing points [10].

$$P = K[R\,T] \qquad K = \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3}$$

3.1 Rotation Estimation

The rotation matrix can be estimated given three orthogonal vanishing points as shown in the following equation [10].

$$R = \begin{bmatrix} \lambda_1(u_1 - u_0)/\alpha & \lambda_2(u_2 - u_0)/\alpha & \lambda_3(u_3 - u_0)/\alpha \\ \lambda_1(v_1 - v_0)/\alpha & \lambda_2(v_2 - v_0)/\alpha & \lambda_3(v_3 - v_0)/\alpha \\ \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix}. \tag{4}$$

Where λ_1 , λ_2 and λ_3 are the scale factors, (u_0, v_0) is the principle point, (u_i, v_i) $i = 1,2,3$ are the three orthogonal vanishing points, and α is the focal length. When only two vanishing points are available, the third vanishing point can be inferred by assuming that the principle point is at the camera center [2].

Equation 4 gives the solution of the rotation matrix for finite vanishing points. However, when the vanishing points are at infinity (lines are parallel in the image space), the solution becomes singular, which cannot be directly given by Equation 4. We derive the equations to compute the rotation matrix for infinite vanishing points using Euler angles. Details please refer to the technique report [11].

The accuracy of the rotation angle depends on the accuracy of the vanishing points. However, for images taken at a skewed angle, only lines close to the camera center will be detected while far lines cannot be detected due to foreshorten effects (Figure 2 left). This cause the vanishing points to be biased, hence the rectified texture of far building surfaces is skewed (Figure 2 right top).

We use a hard constraint technique to solve the bias problem. As shown in Figure 2, we first compute the vanishing point using the detected lines (yellow lines). Then the user interactively indicates a far line (the blue line) as a hard constraint. We define an error function as in Equation 5, where d_i is the distance of the vanishing point to each line, d_h is the distance to the hard constraint line, and w is its weight. We set w to a high value to penalize vanishing points far from the hard constraint line. The LM algorithm is used to find the optimal solution. Figure 2 right bottom shows the texture generated after the hard constraint optimization.

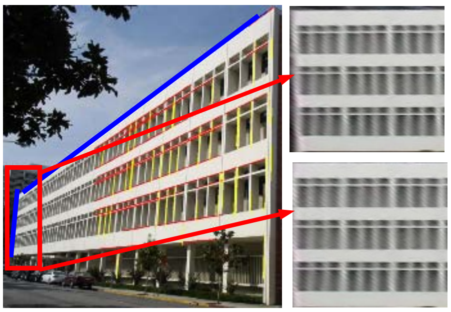


Fig. 2. Hard constraints optimization

$$error = \sum d_i + w d_h \tag{5}$$

3.2 Translation Estimation

Given the orientation of the camera, the translation relative to the 3D models can be computed using two 2D to 3D point correspondences [2]. However, due to the physical limitations (such as a camera's small field of view and narrow streets), sometimes only one 3D corner or no corners are visible for a single image, which makes the translation estimation problem an under-constrained problem.

The technical report [11] presents a technique using multiple images at different viewpoints to solve the problem. However, images taken at far-apart viewpoints have color difference and parallax problems, which cause the generated textures to be blurred. This paper presents a new technique using mosaic images taken at almost the same viewpoint to solve the under-constrained pose problem.

There exists a homography between images taken at the same viewpoint or images of planar structures. Four 2D correspondences are enough to compute the homography. Corners can be used to find the initial homography, and more correspondences can be found to refine the results. Figure 4 (a) shows a generated mosaic image [16]. Our pose recovery algorithm does not require the mosaic image to be a full panorama. As long as the mosaic image covers at least two 3D model points, we can recover the pose and generate the rectified textures. Figure 4 (b) shows the automatically rectified texture from the mosaic image using our technique.

A mosaic image has more image lines than a single image, which makes the vanishing point extraction more reliable. It also covers more building corners, which makes the translation estimation for a single mosaic image possible. Furthermore, images takes at the same viewpoint have no parallax, which makes the texture quality better.

4 Update Textures with Videos

Static textures are useful in creating a realistic image without modeling the details, however, they lack the capability to provide the most recent information of the environment. We use videos as a texture resource to continuously update textures to reflect the most up-to-date changes of environment imagery. Given the pose of the video [13], the key issue in updating textures is to dynamically update the base buffer in real time. Updating the textures for all base buffers at the same time is infeasible for a real-time system, so we dynamically detect a visible base buffer (active base buffer), and first update the buffer using a software technique, then accelerate the process with the features of graphics hardware.

4.1 Active Buffer Selection

A two-step strategy is employed to find the current active base buffer. First, we find all the buildings visible from the current camera viewpoint. To speed up this process, bounding boxes are computed for each building. Then the bounding boxes are intersected with the camera frustum to determine their visibility. Among all the visible buildings, we find the closest one to the current viewpoint. In a second step, we compute the angle between the buffer normal and camera optical axis for all four base buffers of the closest building, and set the buffer with the minimum angle as the current active buffer. If all the visible buffers in the closest building exceed an angle threshold, the next closest building is considered. A weighted average of the normalized building distance and buffer view angle is used to select the active buffer in order to ensure the texture quality (Equation 6).

$$w = \alpha * Dist + \beta * Angle \tag{6}$$

Where *Dist* is the minimum distance from the polygons of a building to the current viewpoint, *Angle* is the minimum angle between the buffers' normal and the view direction, and α and β are the weights. *Dist* and *Angle* are normalized respectively using the minimum distance and angle among all the visible buildings.

4.2 Update Base Buffer

We update the base buffers pixel by pixel with a technique called texture painting from video [14]. The key process in painting textures is a 3D model-based image warping process (Figure 3), therefore we need to know the corresponding 3D model point for every pixel in each base buffer to do the image warping. In order to continuously update textures in real time, we pre-compute the depth map for each base buffer, and use it as a lookup table to find the 3D point for every pixel in the buffer.

As a video frame is processed (Figure 3), we first find the active base buffer according to the aforementioned algorithm. For each pixel I_b in the active buffer, we find the corresponding 3D point M based on the depth map. If the point M passed the occlusion test, we project it into the current camera image plane to find the corresponding pixel I_v. We then update the base buffer pixel with the pixel I_v from the video frame. Textures are segmented if moving objects present, and the warped image is registered with the base buffer if the 3D pose is inaccurate. The system can reach up to 12 fps for video size of 256 x 256 (Table 1), given accurate pose.

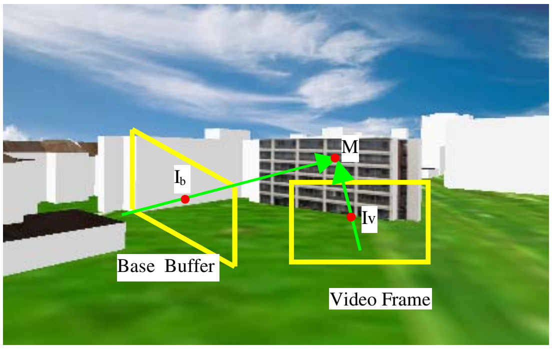


Fig. 3. Update base buffer

4.3 Acceleration with Graphics Hardware

The software implementation achieves real time performance for a small-sized texture image, such as 256 x 256. However, we need to pre-compute and store a depth map for each base buffer, which costs additional memory. For our university campus with 100 buildings, each building has four buffers with a depth map of 512 x 512, more than 400 MB are required to store the depth maps.

Updating high quality textures continuously is hard to achieve real time performance using software implementation, and the storage of depth maps in software implementation is a heavy burden on the system for a large-scale environment. We need a more efficient implementation that uses graphics hardware features to improve the performance and reduce the memory requirement.

Compared to 2D image warping, which can be easily implemented using GPU programming, image warping based on a 3D model is more complex to implement using graphics hardware. If the 3D model is just a plane, the relationship between the base buffer and camera image plane can be described using a homography. However, image warping based on homography is impractical for our purpose because most building geometry is much more complex than a few planes. We cannot afford to segment the video frames and warp them according to different homographies in a time- critical system.

To fully benefit from graphics hardware, we use the technique of projective texture mapping [12]. We first set the current active buffer as the rendering buffer, then render the current scene using projective texture mapping; we solve the visibility problem using a hardware register combination based on a depth map. In order to update the buffer with new textures and keep the old textures, we use an alpha channel to mask out all the pixels outside the camera frustum and those pixels without textures. The process is summarized as the following pseudo code:

1. *Dynamically* detect *the active base buffer.*
2. *Render a depth map from the camera viewpoint. (The* depth *map here is used to solve the visibility problem, not for model and base buffer correspondence.)*
3. *Load the* image *of the current base buffer (with textures already painted) into the rendering buffer.*
4. *Render the scene using projective texture mapping with* the *current video frame as the textures, and solve the visibility problem based on the depth map. Set the alpha value of the pixels without textures or invisible to zero.*
5. *Enable alpha test; only render pixels with alpha value* greater *than zero into the base buffer.*
6. *Read the rendering buffer, and set it as the textures for* the *models corresponding to the active base buffer.*

Table 1. Frame rate and memory requirements of software implementation and graphics hardware acceleration

Method	Frame rate at different sensor resolution			Memory
	256	512	1024	
Software	12 fps	4 fps	1 fps	700 M
Hardware	30 fps	20 fps	10 fps	300 M

We can gain many benefits from the use of graphics hardware. The frame rate increases dramatically, and we can achieve real time for high-resolution textures of 1024 x 1024. Using graphics hardware features, depth map storage for base buffers is not necessary, which saves a great deal of memories. Table 1 compares the frame rate at different sensor resolutions and memory requirements (for a fixed buffer resolution of 512 x 512 and 100 buildings with 40,000 polygons) for software implementation and graphics hardware acceleration on a Dell machine with 3.4 G CPU and NVIDIA quadro 980 graphics card.

5 Results

5.1 Static Textures

A number of static textures are generated using the described method. The 3D models are generated using LiDAR data with a semi-automatic modeling system [1]. Pictures are taken with an un-calibrated camera at different time. The images are stitched as a mosaic Figure 4(a). The pose is automatically estimated, and the mosaic is rectified to generate a high quality texture with resolution 4000 by 3000 pixels (b). Textures generated from mosaic images are often of high quality due to free from color and parallax problems.

We have generated textures for a university campus center area (Figure 5 (a)). More than 100 ground view images were taken to generate 30 façade textures with automatically recovered camera poses. These textures are mapped to the buildings using the base buffers for visualization. Figure 5 (a) shows an overview of the campus environment, and a close up view is shown in (b). Although the 3D models do not have enough details, the rendering results are very realistic thanks to the high quality textures.

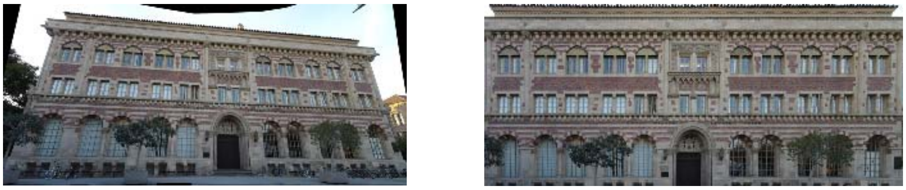


Fig. 4. Generate static textures. (a) A mosaic image; (b) the rectified texture.

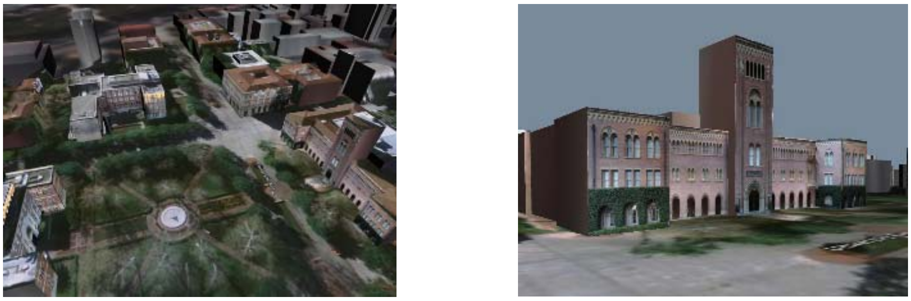


Fig. 5. Rendering results with static textures. (a) An overview; (b) a close up view.

5.2 Update Textures with Videos

We used three real video streams captured from hand-held cameras to paint textures in real time. The motion of the first video stream is mainly rotation, the second translation, and the third both translation and rotation. A portable tracking system using GPS and GYRO (inertial sensor) with a hand held camera [13] was used to collect camera tracking data and video streams. The video streams were projected onto the

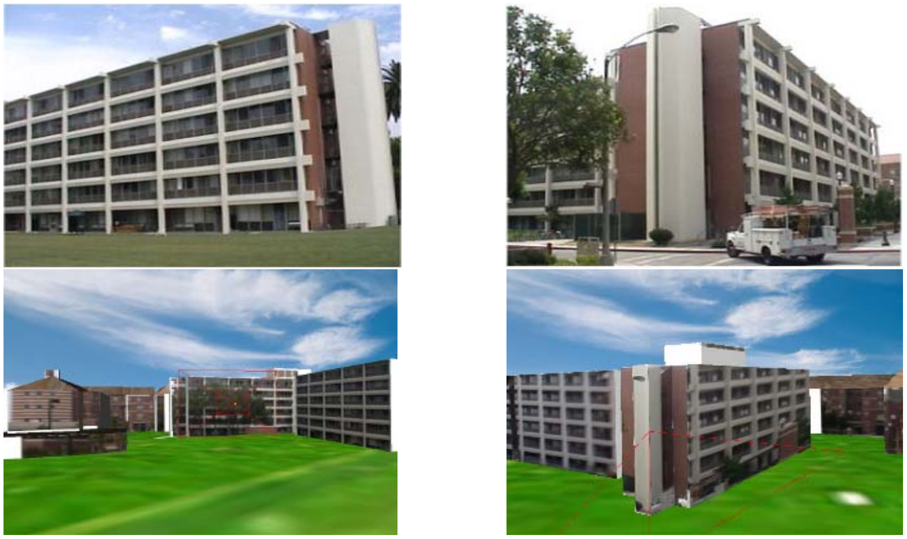


Fig. 6. Real-time painting. Top row: frames from two video streams. Bottom row: painted texture results.

3D models to paint textures using the proposed technique. We only show the results of two video streams due to lack of space (Figure 6). The top row shows two selected frames form the two video sequences, and the bottom row shows the real-time painted dynamic textures. The whole system is fully automatic and real-time (29 fps). Compared with static textures, the textures painted from videos have the advantage of reflecting smoothly varying of images when the viewpoint changes, hence they can visualize the most recent imagery of the environment. An application of painting textures from videos is to visualize multiple videos by projecting them onto 3D models for information visualization and decision-making [13].

6 Conclusion

This paper presents new techniques to generate high quality static and update textures for given urban building models. An efficient way for textures storage of a large-scale environment is presented. Static textures are generated by automatically recover the camera's pose. Furthermore, textures are painted in real time by employing videos as texture resources. Future work will be using the registered images to correct the models and generate more model details.

References

1. You, S., Hu, J., Neumann, U., and Fox, P.: Urban site modeling from LiDAR, CGGM (2003)
2. Lee, S.C., Jung, S. K., and Nevatia, R.: Integrating ground and aerial views for urban site modeling. ICPR (2002)

3. Portilla, J. and Simoncelli, E.P.: A parametric texture model-based on joint statistics of complex wavelet coefficients. IJCV 40(1) (2000) 49–70
4. Efros, A. and Leung, T.: Texture synthesis by non-parametric sampling. ICCV (1999) 1033–1038
5. Efros, A. and Freeman, W.T.: Image quilting for texture synthesis and transfer. Proceedings of SIGGRAPH (2001) 341–346
6. Fruh, C. and Zakhor, A.: Constructing 3D city models by merging aerial and ground views. CGA, 23(6) (2003) 52-61,
7. Stamos, I. and Allen, P.: Automatic registration of 2D with 3D imagery in urban environments. ICCV (2001) 731–736
8. Schodl, A. et al.: Video textures. Proceedings of SIGGRAPH (2000) 489–498
9. O'Rourke. Art gallery theorems and algorithms, Oxford Univ. Press, New York (1987)
10. Cipolla, R., Drummond, T. and Robertson, D.P.: Camera calibration from vanishing points in images of architectural scenes. In BMVC (1999) 382–391
11. Hu, J.: Automatic pose recovery for high-quality textures generation. USC Technique Report 06-874 (2006)
12. Mark, S.: et al. Fast shadows and lighting effects using texture mapping. Siggraph (1978)
13. Neumann, U. et al.: Augmented virtual environments (AVE): dynamic fusion of imagery and 3D models, IEEE Virtual Reality (2003) 61–67
14. Hu, J., You, S. and Neumann, U.: Texture painting from video. Journal of WSCG, ISSN 1213–6972, Volume 13 (2005) 119–125
15. Debevec, P., Taylor C., and Malik, J.: Modeling and rendering architecture from photographs: a hybrid geometry and image based approach, In Proc. of SIGGRAPH (1996)
16. http://www.ptgui.com/

Planar Surface Detection in Image Pairs Using Homographic Constraints

Qiang He and Chee-hung Henry Chu

The University of Louisiana at Lafayette
Center for Advanced Computer Studies
Lafayette, LA 70504-4330, U.S.A.
{qxh3290, cice}@cacs.louisiana.edu

Abstract. Planar surfaces are important characteristics in man-made environments and have been successfully applied to camera calibration and interactive modeling. We develop a method for detecting planes in image pairs under epipolar constraints using planar homographies. In order to extract the whole planes, the normalized cut method is used to segment the original images. We pick those segmented regions that best fit a triangulation of the homography inliers as the detected planes. We illustrate the algorithm's performance using gray-level and color image pairs.

1 Introduction

Planar surfaces encapsulate two-dimensional information of objects, especially in man-made environments. They abound in buildings and architectural structures, in indoor environments, and are found on most manufactured objects. These properties enable the use of planes in many visual computing fields. In the computer vision field, planes have been successfully used for camera calibration [2]. Planar surfaces are also widely applied in augmented reality and interactive modeling [6,7,8]. There are two major problems for planar surface detection from digital images that must be overcome so that they can be used in these applications. One is to compute the planar homography between different images based on entity correspondences. The other is to extract the whole plane from the image background through the planar homography.

In [1], planes are detected based on point and line matchings without camera calibration and 3D scene reconstruction. The searches for planar homographies are implemented through an iterative voting scheme based on the Least-Median-of-Squares principle and the Levenberg-Marquardt algorithm. The possibly mismatched features are handled in a robust manner during the estimation process. The method was tested using on pictures taken of scenes in indoor environments.

Another approach [3] to recovering planar regions of arbitrary position and orientation from plane-induced homographies is to compute dense point matching between two images. These correspondences are subsequently used to estimate the planar homographies, followed by verifying or discarding potential planes. Finally, a region growing process is applied to detect the whole planar region.

In [4], a plane sweep strategy was introduced. In a 3D-space, a 3D-line defines a one-parameter family of planes. Each side of the line is a half plane, which is defined by the 3D line and similarity scores computed over all the views. The inter-image homographies are used to validate and estimate the best-fitting planar surfaces. Line grouping and completion are based on the computed half planes for the final step of plane delineation.

In this paper, we present a method that uses dense matching points with epipolar constraints to detect all potential planes in an image pair. The planar inliers for each fitted plane are joined by a Delaunay triangulation. Because the reliable inliers of each plane tend to be in the plane's interior, they are more useful for labeling than for defining the boundaries of the plane. In order to extract an entire plane, the normalized cut image segmentation method is used to partition the original image. We label those segmented regions with maximum fit to the Delaunay triangulation areas as whole planes. In the following, we describe the method in Section 2, followed by our experimental results in Section 3. We discuss some practical considerations of the implementation in Section 4 and present our concluding remarks in Section 5.

2 Method Description

There is a planar homography among the projections of planar surfaces taken from different viewpoints [5]:

$$\mathbf{x}' = H\mathbf{x} \tag{1}$$

where $\mathbf{x}'$ and $\mathbf{x}$ are the homogeneous coordinates of the projections of the three-dimensional point $\mathbf{X}$ on a world plane in the first and second images, respectively. In general, the homography H is a 3×3 nonsingular matrix and hence its rank is 3. We have a one-to-one point correspondence between the left and right images. If the rank of H is 2, then all the points of the plane in the first image are projected onto a line in the second image. If the rank of H is 1, then all the points of the plane in the first image are projected onto a point in the second image. These degenerate cases of H are not of present interest and are ignored in this paper.

In additional to the homography in Equation (1), the corresponding points from planar homographies also need to satisfy epipolar constraints, viz. $\mathbf{x}'^T F\mathbf{x} = 0$, where F is the fundamental matrix between the image pair. Incorporating planar homography information, we have

$$\mathbf{x}'^T F\mathbf{x} = (H\mathbf{x})^T F\mathbf{x} = \mathbf{x}^T H^T F\mathbf{x} = 0. \tag{2}$$

Similarly, we have

$$\mathbf{x}^T F^T H\mathbf{x} = 0, \tag{3}$$

so that under the epipolar constraint,

$$H^T F + F^T H = 0. \tag{4}$$

Given the fundamental matrix F and three matching points $(\mathbf{x}_i, \mathbf{x}'_i)$, for $i = 1, 2, 3$, we can compute the plane homography induced by the plane defined by three corresponding 3D points as

$$H = \mathbf{e}' \times F - \mathbf{e}'\mathbf{v}^T \tag{5}$$

where $\times$ is the cross product operator, $\mathbf{e}'$ is the right epipole, $\mathbf{v} = M^{-1}\mathbf{b}$, and $M = [\mathbf{x}_1 \;\; \mathbf{x}_2 \;\; \mathbf{x}_3]^T$. The ith element of the 3-vector $\mathbf{b}$ is defined as

$$b_i = (\mathbf{x}_i' \times (A\mathbf{x}_i))^T(\mathbf{x}_i' \times \mathbf{e}')/\|\mathbf{x}_i' \times \mathbf{e}'\|^2. \tag{6}$$

In order to compute the homography, we need to estimate the fundamental matrix and the epipoles. The fundamental matrix is estimated using the normalized 8-point algorithm as described in [5] based on a set of putatively matched 2D points. In our work, Harris corners [12] are extracted as interesting points. The RANSAC strategy [13] is used to fit the fundamental matrix in a robust manner. The epipoles are the right and left null vectors of the fundamental matrix, and can be computed using a singular value decomposition (SVD) of the estimated fundamental matrix.

Next, the homographies can be computed using those matching points that satisfy the epipolar constraints. Each homography can be estimated from Equation (5) robustly using RANSAC. An image may contain more than one plane; we find each of the planes sequentially. After a plane or, equivalently, one homography is verified, we remove all those inliers that correspond to this plane. Then we start to estimate another homography using the rest of the matching points. The process stops when the assumed maximum number of planes is reached or very few inliers are left that satisfy a single homography.

In order to encapsulate the plane surfaces, we compute the Delaunay triangulation [10,11] of the inliers for each detected homography. The Delaunay triangulation of a point set is a collection of edges, each of which has a circle containing the edge's endpoints but not containing any other points. In general, the inliers-based triangulation does not enclose the whole planar surfaces. Consequently, we use some segmentation methods to extract the whole planes. The graph-theoretic normalized cut algorithm [9] captures global information and does not generally result in small partitions. In addition, the normalized cuts are also good at reducing the influence from illusory contour and texture, which is a problem for segmenting architectural buildings. These observations therefore suggest that the normalized cut is a feasible candidate for our requirements. We used normalized cut to segment the planar surfaces; the segments are then label using the Delaunay triangulation meshes.

In summary, the major steps of the algorithm to extract planes are as follows:

1. Robustly estimate the fundamental matrix from matched points based on point normalization and RANSAC.
2. Using RANSAC, estimate the plane homography that fits the most number of matched points that satisfy the epipolar constraint from Step 1.
3. Save the homography and form a Delaunay triangulation of the the inliers of the plane homography.
4. If the maximum number of homography is reached or if the ratio between inliers and total points is less than a threshold, then terminate the search for homographies and go to Step 6.

5. Otherwise, remove inliers of the last homography and consider remaining matched points, ie the outliers of the last homography. Go to Step 2.
6. Partition the original image using the normalized cut image segmentation algorithm. A segmented component that fits the majority of the Delaunay triangulation areas is regarded as a final plane.

The fitting error of homography is defined as the sum of forward-projection as well as backward-projection fitting errors. If the tolerance for the fitting error of homography is too small, a single plane may be fitted by several homographies. On the other hand, if the tolerance is too large, two or more planes may be fitted by one homography. In order to detect all planes, we set the fitting error tolerances to small values. In the final step, we extract and merge the planar surfaces using the normalized cut segmentation.

3 Experiments

The algorithm is implemented and tested on two pairs of real-world images. The first pair is the Minorite Monastery data set, a pair of gray level pictures

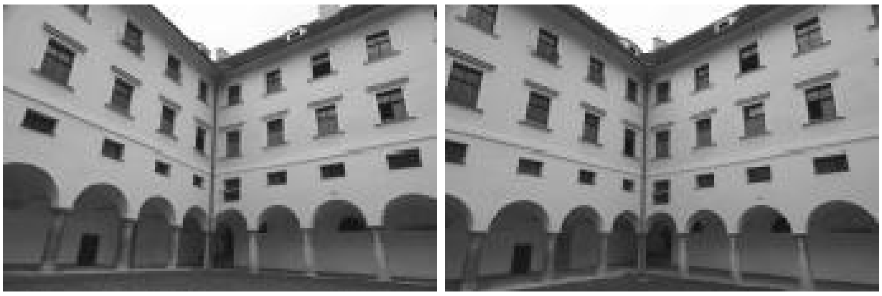


Fig. 1. The original Minorite Monastery image pair

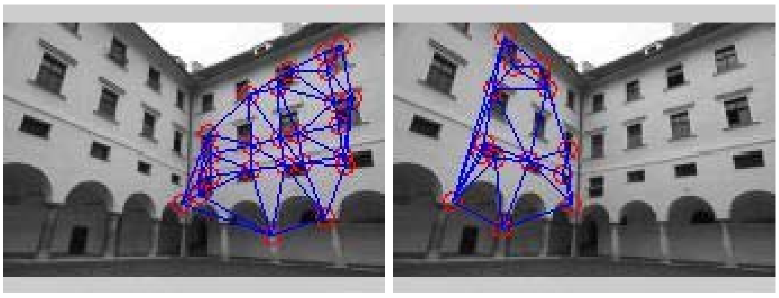


Fig. 2. The first detected homography (*left*) and the second detected homography (*right*)) from the Minorite Monastery data are shown as the Delaunay triangulation of the inliers of each detected planar homography

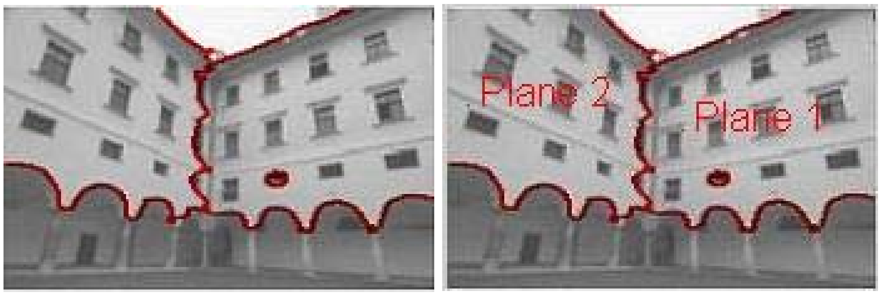


Fig. 3. The final segmented components of the left image using the normalized cut algorithm (*left*) and the detected planes from the Minorite Monastery data (*right*)

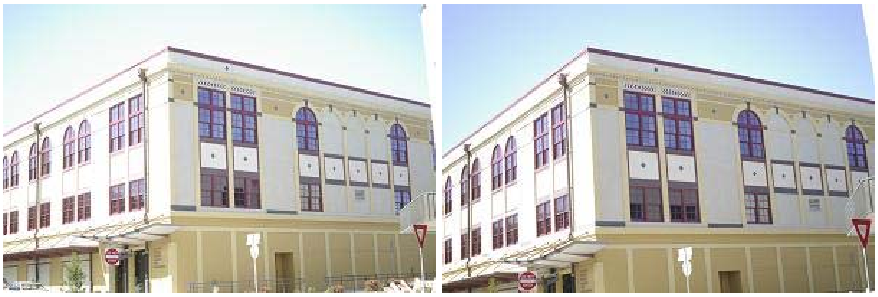


Fig. 4. The original downtown Lafayette (La.) color image pair

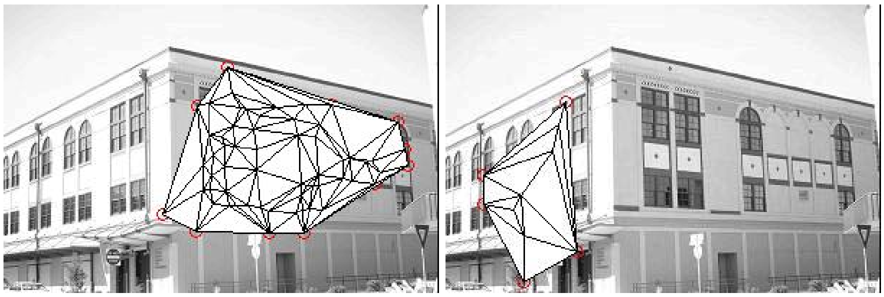


Fig. 5. The first detected homography (*left*) and the second detected homography (*right*)) from the downtown Lafayette data are shown as the Delaunay triangulation of the inliers of each detected planar homography

(Figure 1). Figures 2 and 3 show the results of the Minorite Monastery data set. The inliers and the the Delaunay triangles of two detected homographies are shown in Figure 2. The normalized cut segmentation output and the labeled segments are shown in Figure 3.

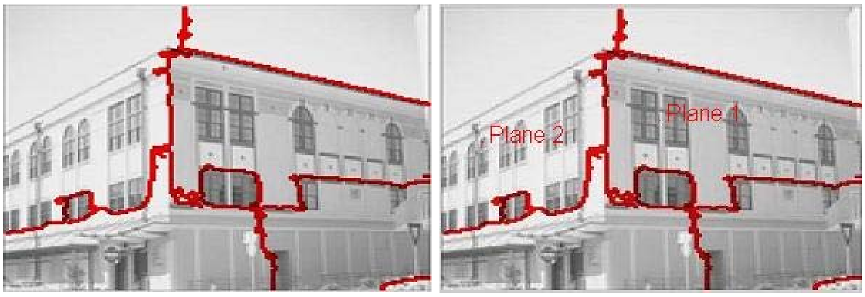


Fig. 6. The final segmented components of the left image using the normalized cut algorithm (*left*) and the detected planes from the downtown Lafayette data (*right*)

The second pair are color images of a building in downtown Lafayette, La. (Figure 4). The Harris corners are detected from the intensities of the images and the plane homographies are estimated from matched pairs of the Harris corners. The inliers and the the Delaunay triangles of two detected homographies are shown in Figure 5. The normalized cut segmentation output and the labeled segments are shown in Figure 6.

4 Discussions

We note that the major planes have been successfully extracted from the images in the experiments described in Section 3. The performance of plane detection depends directly on the accuracy of the homography computation, as shown by comparing the better result in Figure 3 to the result in Figure 6. Furthermore, our experiments show that the normalized cut algorithm can produce small blocks in complicated pictures, as shown in Figure 6.

In practice, we find that the following three problems have to be addressed in plane detection. Under a pure rotation of the camera, there is a homography from the first image to the second one. This homography is stronger than any inter-image planar homography since all image points satisfy this homography. We should discard it in order to detect those inter-image planar homographies. Additionally, if the rank of a homography is less than 3, it is degenerate. We should check this kind of homographies. Finally, the performance of normalized cut image segmentation needs to be improved in future work.

4.1 Homography from Pure Rotation

If the only motion of camera is a pure rotation about its center, then there is a plane homography between the left and the right images. Define the projection matrices of the left image and the right image as, respectively,

$$\mathbf{P}_1 = K[\, I \;\mid -\tilde{\mathbf{C}}]$$

and

$$\mathbf{P}_2 = KR[\ I \ \ | \ -\tilde{\mathbf{C}}],$$

where the nonsingular matrix K is the camera calibration matrix associated with its internal parameters, R describes the rotation of the camera relative to the world coordinate system, and $\tilde{\mathbf{C}}$ is the coordinates of the camera center in the world coordinate system. Then given a world point $\mathbf{X}$ and its projections $\mathbf{x}$ and $\mathbf{x}'$ on the left and right images, we have $\mathbf{x} = \mathbf{P}_1\mathbf{X} = K[\ I \ \ | \ -\tilde{\mathbf{C}}]\mathbf{X}$ and $\mathbf{x}' = \mathbf{P}_2\mathbf{X} = KR[\ I \ \ | \ -\tilde{\mathbf{C}}]\mathbf{X}$, so that

$$\mathbf{x}' = KR[\ I \ \ | \ -\tilde{\mathbf{C}}]\mathbf{X} = KRK^{-1}K[\ I \ \ | \ -\tilde{\mathbf{C}}]\mathbf{X} = KRK^{-1}\mathbf{x} = H\mathbf{x},$$

where $H = KRK^{-1}$ defines a plane homography between the entire left image and the entire right image.

Under this scenario, we should discard this major homography in order to detect inter-image homographies of the planar surfaces. In our implementation, we segment the original images first and only consider the potential homographies within different segmented components. If a homography is verified to exist in some component, then this component is regarded as a planar surface.

4.2 Homography Degeneracy

As observed in Section 2, the homography H is in general nonsingular and hence its rank is 3. We consider the cases when the rank of H is less than 3 in the following. We can describe H using its row vectors:

$$H = \begin{bmatrix} \mathbf{h}_1^T \\ \mathbf{h}_2^T \\ \mathbf{h}_3^T \end{bmatrix}$$

where $\mathbf{h}_1^T$, $\mathbf{h}_2^T$, and $\mathbf{h}_3^T$, are the three row vectors of H.

Case 1. Rank of H is 2.

In this case, $\mathbf{h}_3$ is a linear combination of $\mathbf{h}_1$ and $\mathbf{h}_2$, so that we have $\mathbf{h}_3 = c_1\mathbf{h}_1 + c_2\mathbf{h}_2$ for some constants c_1 and c_2. Then we have

$$\mathbf{x}' = H\mathbf{x} = \begin{bmatrix} \mathbf{h}_1^T \\ \mathbf{h}_2^T \\ \mathbf{h}_3^T \end{bmatrix} \mathbf{x} = \begin{bmatrix} \mathbf{h}_1^T\mathbf{x} \\ \mathbf{h}_2^T\mathbf{x} \\ \mathbf{h}_3^T\mathbf{x} \end{bmatrix} = \begin{bmatrix} \mathbf{h}_1^T\mathbf{x} \\ \mathbf{h}_2^T\mathbf{x} \\ (c_1\mathbf{h}_1 + c_2\mathbf{h}_2)^T\mathbf{x} \end{bmatrix}.$$

Define a line $\mathbf{l}' = [c_1 \quad c_2 \quad -1]^T$ in the right image. Then,

$$\mathbf{x}'^T\mathbf{l}' = \begin{bmatrix} \mathbf{h}_1^T\mathbf{x} & \mathbf{h}_2^T\mathbf{x} & (c_1\mathbf{h}_1 + c_2\mathbf{h}_2)^T\mathbf{x} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ -1 \end{bmatrix} = 0.$$

Therefore, all the points of the plane in the first image are projected onto a line in the second image.

Case 2. Rank of H is 1.

In this case, both $\mathbf{h}_1$ and $\mathbf{h}_2$ are multiples of $\mathbf{h}_3$. That is, $\mathbf{h}_1 = c_1\mathbf{h}_3$ and $\mathbf{h}_2 = c_2\mathbf{h}_3$ for some constants c_1 and c_2. We then have

$$\mathbf{x}' = H\mathbf{x} = \begin{bmatrix} \mathbf{h}_1^T \\ \mathbf{h}_2^T \\ \mathbf{h}_3^T \end{bmatrix} \mathbf{x} = \begin{bmatrix} \mathbf{h}_1^T\mathbf{x} \\ \mathbf{h}_2^T\mathbf{x} \\ \mathbf{h}_3^T\mathbf{x} \end{bmatrix} = \begin{bmatrix} c_1\mathbf{h}_3^T\mathbf{x} \\ c_2\mathbf{h}_3^T\mathbf{x} \\ \mathbf{h}_3^T\mathbf{x} \end{bmatrix} = (\mathbf{h}_3^T\mathbf{x}) \begin{bmatrix} c_1 \\ c_2 \\ 1 \end{bmatrix}.$$

Noting that $\mathbf{h}_3^T\mathbf{x}$ is a scalar, we get $\mathbf{x}' = [c_1 \;\; c_2 \;\; 1]^T$. Therefore, all the points of the plane in the first image are projected onto a point in the second image.

The above two degenerate cases of H are not of interest to our present discussion. Hence, we can check and then ignore these cases. Note that when the rank of H is 1 or 2, the vectors $\mathbf{h}_1$, $\mathbf{h}_2$, and $\mathbf{h}_3$ are collinear. To detect the degeneracy of H, it suffices to check whether $\mathbf{h}_1^T(\mathbf{h}_2 \times \mathbf{h}_3) = 0$, where $\times$ is the vector cross product.

4.3 Segmentation Improvement

We used the normalized cut algorithm to segment the image. Some of the segments are then declared planar surfaces if they contain feature points that are inliers of detected homographies. During the segmentation process, the number of detected homographies can be used as the number of potential clusters in the normalized cut algorithm. In practice, we use a number that is slightly larger than the number of homographies to accommodate other nonplanar surfaces.

Although in theory the normalized cut algorithm does not result in small segments, in practice this is not always the case. Improvements of the image segmentation step remain a direction for our future work.

5 Concluding Remarks

We provide a plane detection method that incorporates epipolar information between a pair of images. All potential planes that satisfy plane homographies are first detected. A Delaunay triangulation process is used to link the planar inliers for each detected plane. Finally, we applied the normalized cut algorithm to segment the original images and obtain the plane regions that contain the inliers of plane homographies.

Acknowledgments

We sincerely thank Dr. Konrad Schindler for courtesy permission to use the Minorite Monastery pictures. This work was supported in part by the Louisiana Governor's Information Technology Initiative.

References

1. M. Lourakis, A. Argyros, and S. Orphanoudakis, "Detecting planes in an uncalibrated image pair," in *Proc. BMVC'02*, volume 2, 2002, pp. 587–596.
2. P. Sturm and S. Maybank, "On plane-based camera calibration: A general algorithm, singularities, applications," in *IEEE Conf. Computer Vision and Pattern Recognition*, 1999, pp 432–437.
3. K. Schindler, "Generalized use of homographies for piecewise planar reconstruction," in *Proc. 13th Scandinavian Conference on Image Analysis (SCIA'03)*, Gotenborg, Sweden, 2003.
4. C. Baillard and A. Zisserman, "A plane-sweep strategy for the 3d reconstruction of buildings from multiple images," in *Proc. 19th ISPRS Congress and Exhibition,* 2000, pp. 56–62.
5. R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision,* second edition, Cambridge University Press, 2003.
6. A. Criminisi, I. Reid, and A. Zisserman, "A plane measuring device," *Image and Vision Computing*, vol. 17, pp. 625–634, 1999.
7. G. Simon, A. Fitzgibbon, and A. Zisserman, "Markerless tracking using planar structures in the scene," in *Proc. Int. Symp. Augmented Reality*, 2000.
8. I. Zoghlami, O. Faugeras, and R. Deriche, "Using geometric corners to build a 2D mosaic from a set of images," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1997, pp. 420–425.
9. J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 888–905.
10. S. W. Sloan and G. T. Houlsby, "An implementation of Watson's algorithm for computing 2-D Delauney triangulations," *Advanced Engineering Software,* vol. 6, 1984.
11. J. O'Rourke, *Computational Geometry in C*, 1st edition, Cambridge University Press, 1994.
12. C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. 4th Alvey Vision Conference*, 1988, pp. 147–151.
13. M.A. Fischler and R.C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, pp. 381–395, 1981.

Robust Quality-Scalable Transmission of JPEG2000 Images over Wireless Channels Using LDPC Codes

Abdullah Al Muhit and Teong Chee Chuah

Faculty of Engineering, Multimedia University
Jalan Multimedia, Cyberjaya, Selangor 63100, Malaysia
{muhit, tcchuah}@mmu.edu.my

Abstract. A new error-resilient JPEG2000 wireless transmission scheme is proposed. The proposed scheme exploits the 'progressive by quality' structure of the JPEG2000 code-stream and takes into account the effect of channel errors at different quality layers in order to protect the coded bit-stream according to channel conditions using multi-rate low-density parity-check (LDPC) codes, leading to a flexible joint source-channel coding design. The novelty of this adaptive technique lies in its ability to truncate the less important source layers to accommodate optimal channel protection to more important ones to maximize received image quality. Results show that the proposed scheme facilitates considerable gains in terms of subjective and objective quality as well as decoding probability of the retrieved images.

Keywords: JPEG2000, semi-random LDPC codes, unequal error protection.

1 Introduction

The JPEG2000 image coding standard was introduced recently to overcome the limitations of the existing JPEG standard [1]. JPEG2000 provides a wide range of remarkable features, including improved coding efficiency, scalability by quality or resolution, and error resilience features. With the proliferation of broadband communication systems and the growing popularity of handheld devices, the JPEG2000 standard is promising for deployment in wireless imaging applications.

1.1 Wireless Image Transmission Problem

The wireless links are subject to random bit errors caused by channel impairments. Thus, in order for power-constrained devices to achieve high performance over noisy wireless channels, the JPEG2000 code-stream needs to be protected against channel impairments.

In Part 1 of the standard, error resilience is partially achieved by the inclusion of resynchronisation markers, the coding of data in relatively small independent blocks, and the provision of error detection and concealment mechanism within each block. Apparently, an error correction capability is non-existent in the error resilience mode of the standard. Furthermore, these tools do not apply to the image headers (Main header, Tile-part(s) header(s) and packet(s) header(s)) that are the most crucial parts of the code-stream. In noisy channels, errors within packets[1] degrade image quality, whereas errors in the headers can incur severe degradation in image quality as well as decoding failures even in the presence of error resilience tools [2]. Hence, the transmission of pictorial information over unreliable heterogeneous networks has necessitated the need for an additional error protection mechanism to effectively reduce the decoding failure of JPEG2000 imagery and enhance the image quality.

The automatic repeat request (ARQ) is one such way of providing error-free transmission where the packets affected by bit-errors are retransmitted. However, the additional delay introduced by ARQ is unacceptable for real-time applications. The additional bandwidth requirement for ARQ is not viable as well. On the contrary, forward error correction (FEC) based systems do not require retransmission and are thus more preferable over ARQ.

1.2 Related Work

Many FEC based joint source-channel coding (JSCC) schemes have been proposed in order to overcome the decoding failure problem and improve the quality of JPEG2000 images over Binary Symmetric Channels (BSC) [3] as well as Rayleigh fading channels [2]. Banister *et al.* [3] protect different packets of the JPEG2000 bit-stream with a 16-bit Cyclic Redundancy Check (CRC) outer code and an inner turbo code. The optimization problem, in this case, was solved using a high complexity brute-force search algorithm. In a related work [6], Boulgouris *et al.* proposed a robust embedded wavelet based coding scheme in conjunction with FEC based on CRC and rate-compatible punctured-convolutional (RCPC) codes. In [2], Sanchez *et al.* provide different levels of protection to different packets of JPEG2000 images based on their inclusion of bits from the most significant bit planes. Channel protection is achieved here by concatenating a CRC outer coder and an inner RCPC coder. This technique, however, does not assure high decoding probability of images at lower transmission power.

1.3 Our Contributions

In this paper, we outline a robust and flexible JPEG2000 wireless channel protection scheme using LDPC codes. The adaptive rate allocation scheme obtains all necessary information about the source from the JPEG2000 compliant encoder. Semi-random LDPC codes [9] of multiple rates are used for channel coding. The source and channel code rates are optimized in an iterative fashion. Two popular models are used to evaluate the performance of our robust image transmission schemes: 1) BSC and 2) Rayleigh fading channel. Unlike many conventional rate allocation schemes, our proposed method does not impose any restrictions on the source-coding rate. This scheme is able to optimize channel protection for scalable image sources when the source is compressed at a rate less than, equal to or greater than the total available transmission rate. This is due to its unique ability to truncate less important layers when necessary. The proposed scheme can be readily incorporated with any block or convolutional channel coding scheme for any channel model.

[1] A collection of data from the compressed image. Packets, in the context of JPEG2000 are different from network packets.

The rest of the paper is organized as follows. Section 2 describes the preliminaries of JPEG2000 and semi-random LDPC codes. Section 3 presents the proposed method. The performance of the proposed scheme is evaluated in section 4. Finally, we conclude the paper in section 5.

2 Preliminaries

2.1 Quality-Progressive Structure of JPEG2000

JPEG2000 images are very good example of highly scalable data sources. In a quality progressive bit-stream, dependencies exist between successive quality layers. During the coding process, as shown in Fig. 1, the image is (optionally) partitioned into a number of rectangular non-overlapping blocks called tiles. Then the discrete wavelet transform (DWT) is applied on each tile to analyze it into a number of subbands at different resolution levels. Following that, the subbands in each resolution level are further partitioned into code-blocks. Each code-block's embedded bit-stream is distributed across a number of quality layers, L_i for $1 \le i \le \Lambda$. Here, Λ is the number of maximum quality layers. The Embedded Block Coding with Optimized Truncation of the embedded bit-streams (EBCOT) algorithm [8] is used to determine the contributions from each code-block to different layers.

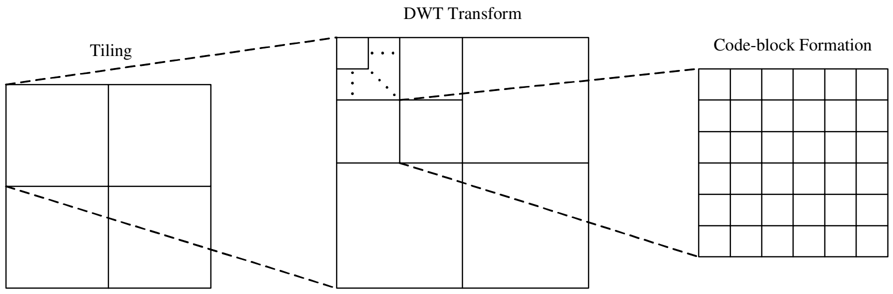


Fig. 1. Data flow in JPEG2000 encoding process

The first quality layer L_1 contains significant code-block contributions. Subsequent layers L_i contain additional contributions from each code-block. However, in some of the subsequent layers the contribution from some code-blocks may be empty. We can express this dependency through a simple linear chain relationship: $L_1 \prec L_2 \prec L_3 \prec \ldots\ldots \prec L_{\Lambda-1} \prec L_\Lambda$; it means that layer L_1 must be decoded before layer L_2 can be used and so on. If one intermediate layer is discarded, the subsequent layers have to be discarded as well. Contribution to each layer from code-blocks belonging to a specific resolution level is grouped together in packets for inclusion in the final bit-stream. A quality progressive code-stream is constructed by sequencing all packets corresponding to quality layer L_1 followed by all packets corresponding to L_2 and so forth. The packets belonging to a quality layer L_i are formed in an order that follows the resolution levels [4]. Since the initial layers contain initial coding passes from all the code-blocks, they form the basis of an image whereas subsequent quality layers impart enhancement on the image. Hence, any error in the initial layers can degrade the quality of the reconstructed image severely. Thus, they require more protection during transmission.

2.2 Semi-random LDPC Codes

Low-density parity-check (LDPC) codes, invented by Gallager [5] are a class of linear block codes that provide near-capacity performance while admitting low-complexity iterative probabilistic decoding [5]. Its $(n-k)\times n$ parity check matrix, H, has a low density of 1's. Here n is the block length and k is the message length. Thus the code rate can be calculated as k/n. The parity check matrix in a semi-random LDPC code [9] can be given as $[H^p, H^d]$. Here H^p is a $(n-k)\times(n-k)$ square matrix and H^d is a semi-randomly generated $(n-k)\times k$ matrix with a column weight of t and row weight of $kt/(n-k)$. H^d is used to encode the message. The parity check matrix H is used for iterative decoding of the received codeword with prior information. The prior information is derived from the received codeword based on the channel model. Due to its capacity approaching performance [5], LDPC-based coding systems are promising for deployment in future communication standards.

3 Methodology

Consider an image source to be transmitted with a total transmission rate of R_T. In our JSCC approach, we have to choose the set of source coding rates, $R_S = \{R_{S1}, R_{S2}, \ldots, R_{S\Lambda}\}$ and the set of channel coding rates, $R_C = \{R_{C1}, R_{C2}, \ldots, R_{C\Lambda}\}$ in order to maximize the overall PSNR (peak signal to noise ratio) of the source, subject to the total rate constraint. Here, $R_{S,i}$ and $R_{C,i}$ are the source and channel coding rate associated with layer L_i. Let Q_i denote the increase in total PSNR of the image due to the inclusion of layer L_i. So the objective is to maximize the following:

$$Q_T = \sum_{i=1}^{\Lambda} Q_i P_i\left(R_{C,i}\right) \tag{1}$$

Here, $P_i\,(R_{C,i})$ is the probability of recovering layer L_i with code rate $R_{C,i}$ assigned to it under a specific channel model. $R_{C,i}$ can be chosen from a set of available code rates $\{R_j\}_{1\le j\le N}$, where N is the total number of code rates. Our optimization problem is to maximize the total PSNR given in (1), subject to the following transmission rate constraint:

$$R_{eff} = \sum_{i=1}^{\Lambda} \frac{R_{S,i}}{R_{C,i}} \le R_T \tag{2}$$

The quality progressive source produced by JPEG2000 is generally convex, i.e. the PSNR-rate characteristic curve is convex as shown in Fig. 2. That is,

$$\frac{Q_1}{R_{S,1}} \geq \frac{Q_2}{R_{S,2}} \geq \cdots \geq \frac{Q_A}{R_{S,A}} \tag{3}$$

Here $S_S(i) = \frac{Q_i}{R_{S,i}}$ is defined as the slope of L_i. Due to this convex characteristic and linear dependency of quality layers, the optimal solution should ensure that recovery of layer L_i will guarantee the recovery of layers L_1 till L_{i-1}. Thus, we can impose another constraint:

$$R_{C,1} \leq R_{C,2} \leq \ldots \leq R_{C,A} \tag{4}$$

It can be noted here that smaller values of code rate mean stronger channel protection and vice versa.

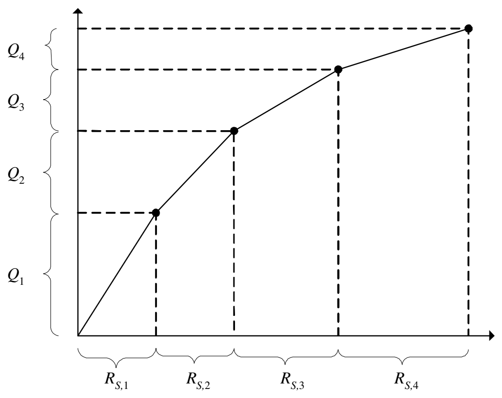


Fig. 2. Rate-PSNR characteristic for a convex source

The PSNR contribution Q_i can be calculated from the JPEG2000 encoder as part of the optimum rate allocation used in the bit plane coding and bit-stream organization [8]. The probability of receiving layer L_i assuming $R_{C,i} = R_j$ with no error can be expressed as:

$$P_i\left(R_j\right) = \left(1 - e_j\right)^{l_i} \tag{5}$$

Here l_i is the length of layer L_i and e_j is the residual bit error rate of R_j. For a BSC, e_j can be found as a function of BER, R_j and n; i.e. $e_j = f(\text{BER}, R_j, n)$ where BER is the cross-over probability or bit error rate and n is the block length of the code. For a BSC, BER reflects the channel condition. And similarly for Rayleigh fading channels, e_j can be calculated as $e_j = f(E_b/N_0, f_D, R_j, n)$ where E_b/N_0 is the ratio of energy per information bit and noise power density and f_D is the doppler frequency. In the case of a Rayleigh fading channel, E_b/N_0 corresponds to the channel condition. We can tabulate e_j from extensive Monte Carlo simulations, for each permissible code rate, and for different BER's and also for various E_b/N_0 and f_D values.

The aim of this novel rate allocation scheme is to enable truncation of less important source layers in order to provide more protection to more important ones when necessary. It is also desirable that our scheme is capable of optimizing channel protection to a scalable source compressed at any rate. Thus, we introduce a special channel code rate $R_j = \infty$; it implies that the corresponding layer is discarded for transmission. We also include another special code rate $R_j = 1$, which means that the relevant layer is transmitted without any channel protection. The first case comes into usage when the channel condition deteriorates drastically whereas the second one is useful if the channel quality improves significantly.

Having dealt with all the decision variables, let us restate the optimization problem: our task is to maximize the objective function in (1) subject to the constraints given by (2) and (4). Alternatively, if we can find a solution using (1) and (2), and later show that the derived solution satisfies the constraint in (4), this would be a feasible solution to our problem. Therefore, we ignore constraint in (4).

We convert this constrained nonlinear optimization problem to an unconstrained optimization problem using Lagrange multiplier [7]. Let $\lambda > 0$ be the Lagrange multiplier; specifically, let $Q_T(\lambda)$ and $R_{eff}(\lambda)$ denote the expected total PSNR and effective transmission rate associated with the set $\{R_{C,i}(\lambda)\}_{1\le i\le \Lambda}$ which maximizes the functional:

$$\begin{aligned} J(\lambda) &= Q_T(\lambda) - \lambda R_{eff}(\lambda) \\ &= \sum_{i=1}^{\Lambda} Q_i P_i(R_{C,i}(\lambda)) - \lambda \sum_{i=1}^{\Lambda} \frac{R_{S,i}}{R_{C,i}(\lambda)} \end{aligned} \tag{6}$$

Thus if we can find λ such that $R_{eff}(\lambda) = R_T$, the set $\{R_{C,i}(\lambda)\}_{1\le i\le \Lambda}$ will form an optimal solution to our constrained problem. In practice, the discrete nature of the problem prevents us from finding λ such that $R_{eff}(\lambda)$ is exactly equal to R_T. However, if many source layers are used with small spacing between them, the rate allocation is close to optimal. So if we can find the smallest value of λ so that $R_{eff}(\lambda) \le R_T$, that would be the optimal solution to this problem.

The unconstrained maximization problem can be decomposed into a collection of Λ separate maximization problems (since it is a separable function). Thus, we need to find the set $\{R_{C,i}(\lambda)\}_{1\le i\le \Lambda}$ which maximizes:

$$J_i(\lambda) = Q_i P(R_{C,i}(\lambda)) - \lambda \frac{R_{S,i}}{R_{C,i}(\lambda)} \tag{7}$$

In a similar optimization problem in EBCOT [4, pp. 339-348], it is shown that feasible solutions lie on the convex hull. Hence, our feasible solution $\{R_{C,i}(\lambda)\}_{1\le i\le \Lambda}$ can be found from the vertices of convex hull of the $P(R_j)$ vs. $1/R_j$ characteristic as shown in Fig. 3. Let 0, 1,…, m be the vertices of the convex hull of the above characteristic. Hence, the slopes of the vertices can be found as

$$S_C(m) = \frac{P(R_m) - P(R_{m-1})}{1/R_m - 1/R_{m-1}}, m > 0 \tag{8}$$

We also denote $S_C(0) = \infty$. Hence, $S_C(0) \ge S_C(1) \ge \cdots \ge S_C(m)$. Consequently, a feasible solution to our problem can be found from

$$\begin{aligned}
R_{C,i}(\lambda) &= \{\max R_m \mid Q_i P(R_m) - \lambda \frac{R_{S,i}}{R_m} \geq Q_i P(R_{m-1}) - \lambda \frac{R_{S,i}}{R_{m-1}}\} \\
&= \{\max R_m \mid Q_i P(R_m) - Q_i P(R_{m-1}) \geq \lambda \frac{R_{S,i}}{R_m} - \lambda \frac{R_{S,i}}{R_{m-1}}\} \\
&= \{\max R_m \mid Q_i (P(R_m) - P(R_{m-1})) \geq \lambda R_{S,i} (\frac{1}{R_m} - \frac{1}{R_{m-1}})\} \\
&= \{\max R_m \mid \frac{P(R_m) - P(R_{m-1})}{1/R_m - 1/R_{m-1}} \geq \lambda \frac{R_{S,i}}{Q_i}\} \\
&= \{\max R_m \mid S_C(m) \geq \frac{\lambda}{S_S(i)}\}
\end{aligned} \tag{9}$$

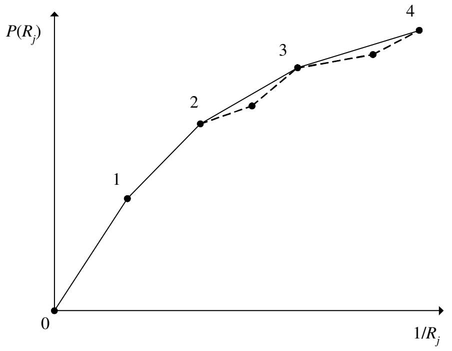


Fig. 3. Vertices of the convex hull of $P(R_j)$ vs. $1/R_j$ characteristic. Vertices connected using dash line are not elements of the convex hull set.

Now, let us show that this solution fulfills the constraint in (4). Due to source convexity, $\frac{1}{S_S(i)} \leq \frac{1}{S_S(i+1)}$; thus we have [10]:

$$\{R_m \mid S_C(m) \geq \frac{\lambda}{S_S(i+1)}\} \supseteq \{R_m \mid S_C(m) \geq \frac{\lambda}{S_S(i)}\} \tag{10}$$

Hence, it can be shown that

$$R_{C,i}(\lambda) = \{\max R_m \mid S_C(m) \geq \frac{\lambda}{S_S(i)}\} \leq \{\max R_m \mid S_C(m) \geq \frac{\lambda}{S_S(i+1)}\} = R_{C,i+1}(\lambda) \tag{11}$$

To summarize, the following iterative algorithm is proposed:

1) Initialize $R_S \geq R_T$ and $\{R_{C,i}\}_{1 \leq i \leq \Lambda} = \infty$.
2) Compress the image with rate = R_S with Λ layers. An internal heuristic determines the lower bound and logarithmically/monotonically spaces the layer rates $\{R_{S,i}\}_{1 \leq i \leq \Lambda}$ over the range [8]. The logarithmic/monotonic spacing

is always convex. The spacing can also be done with an arbitrary or fixed increment. However, the prerequisite for this algorithm is that the source must have a convex rate vs. PSNR (or distortion) characteristic.

3) Choose the parameter, $\lambda > 0$, which controls the trade-off between the expected PSNR, Q_T, and effective transmission rate, R_{eff}. The selection of initial λ plays an important role on the total number of iterations needed to find the optimal λ. We empirically choose $\lambda = \max(S_S)$. Experiments show that this selection reduces the iteration number.
4) For layer i, choose $R_{C,i}(\lambda)$ using equation (9).
5) Calculate $R_{eff}(\lambda)$. If $R_{eff}(\lambda) > R_T$, go to step 7. Otherwise, $\{R_{C,i}(\lambda)\}_{1 \le i \le \Lambda}$ is a feasible solution . Go to step 6.
6) Repeat step 4 and 5 for the next layer.
7) Change λ to obtain $R_{eff}(\lambda) \cong R_T$. An efficient bisection algorithm [7] can be used to find the optimal λ. Once the optimal value has been reached, further alteration of λ will not yield a better solution. Thus, the set of channel code rates associated with optimal λ will be the optimal solution.

In this proposed scheme, channel protection is assigned on a layer-by-layer basis. Following that, the rate constraint is checked. If the assignment exceeds the constraint, we truncate some of the final layers to match the available transmission rate. λ is altered in order to match the rate constraint as close as possible. This iterative process continues until further change in λ does not have any effect. Apart from the layers, the main header of the image is protected with the highest protection available at any channel condition. This is due to the fact that any error in the main header will cause severe degradation of image quality and even decoding failures. However, allocating the highest level of protection to the main header will cause very negligible effect on the rate constraint due to its very short length. Nevertheless, this issue can be addressed to further optimize the transmission scheme.

4 Performance Evaluation

A 512×512 gray-level (8-bit monochrome) 'Lena' image has been used as the test image. 20 quality-layers were generated using lossless compression. A set of semi-random LDPC codes with rates {0.25, 0.33, 0.4, 0.5, 0.66, 0.8, 0.86, 0.89, 0.92, 0.95} and $n = 4000$ was constructed using the method described in [9]. The rate allocation algorithm is then applied to attain different transmission rates over various channel conditions. After rate allocation is performed, each channel condition has been tested with 100 independent trials. The LDPC decoder exits the iterative decoding process once all parity-check equations are satisfied; otherwise the iteration continues until a maximum of 100 iterations.

4.1 Results in BSCs

Error rate in the region of 0.001 and 0.0001 are commonly observed in wireless channels. In worst-case scenarios, it can be as high as 0.01 [6]. Thus, the robustness of the proposed scheme is evaluated at BER = 0.01 and 0.001 for R_T = 0.25, 0.5 and 1 bit per pixel (bpp). The results are compared with those of Boulgouris *et al.* [6] and

Banister *et al.* [3] in Fig. 4 (in terms of average PSNR). Comparisons are strictly made on the basis of equal bandwidth.

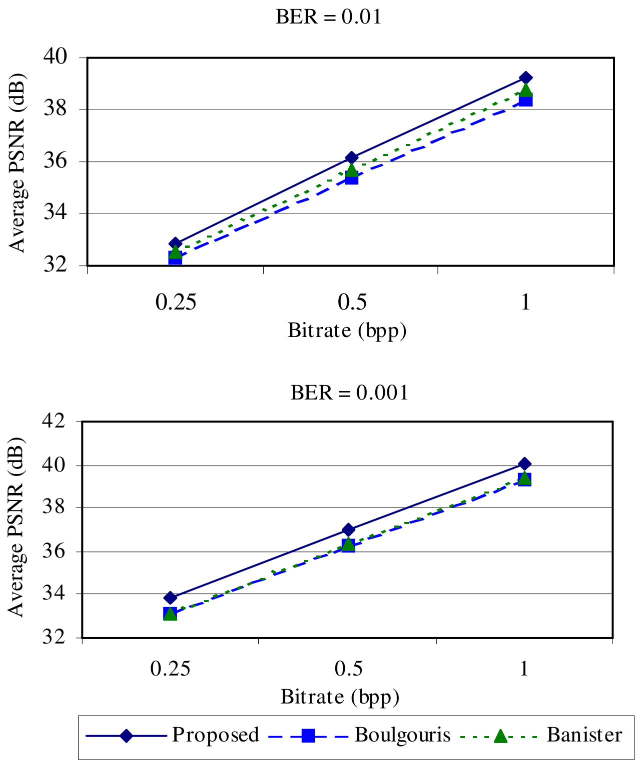


Fig. 4. Comparison of 'Lena' image transmission over BSC

At high BERs (≈ 0.01), our proposed scheme outperforms the above schemes by about 0.4~0.7 dB in PSNR. At lower BERs (≈ 0.001), the gap increases to 0.75 dB in our favour. It is clear that our scheme is superior in worse channel conditions. We can also note that the gap between this scheme and others' widens as the channel condition improves. Progressive nature of the proposed scheme is also demonstrated in Fig. 4. It can be inferred that at any given channel BER the quality of the reconstructed image increases almost linearly with the total transmission rate. It is worth mentioning that due to maximum protection for the headers, very high successful decoding probabilities in the region of 0.99~1.00 were achieved in both BER cases. In subjective comparison over BSCs, the proposed scheme produces images with similar or slightly better quality (as suggested by the objective results mentioned earlier) than those in [3] and [6].

4.2 Results in Rayleigh Fading Channels

We also simulate our transmission scheme over a multi-path Rayleigh fading channel using BPSK modulation with a carrier frequency of 900 MHz, a data rate of 15 kbps, and a maximum Doppler shift of 3 Hz. The performance has been evaluated for R_T = 0.25, 0.5 and 4.41 bpp, by varying the average E_b/N_0 from 10 dB to 15 dB. Fig. 5 presents the average PSNR results in comparison with those of Sanchez *et al.* [2].

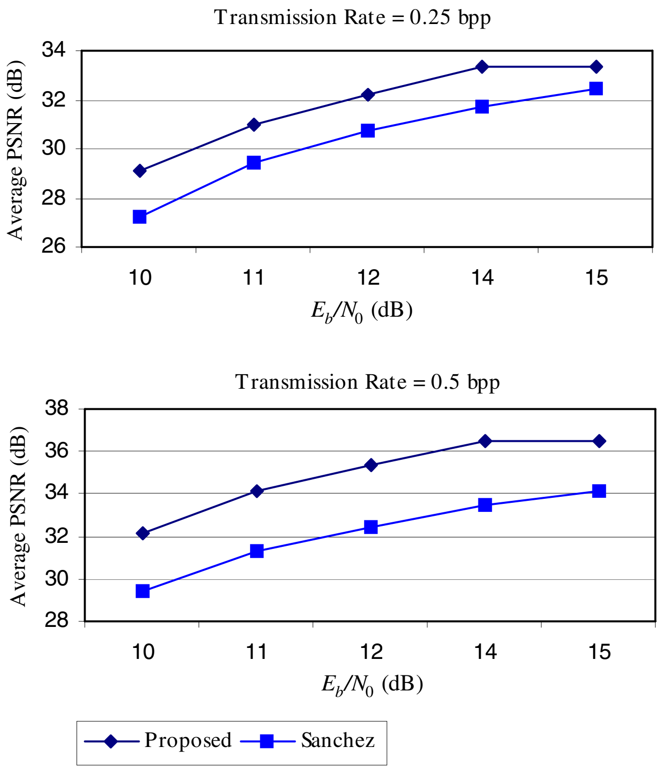
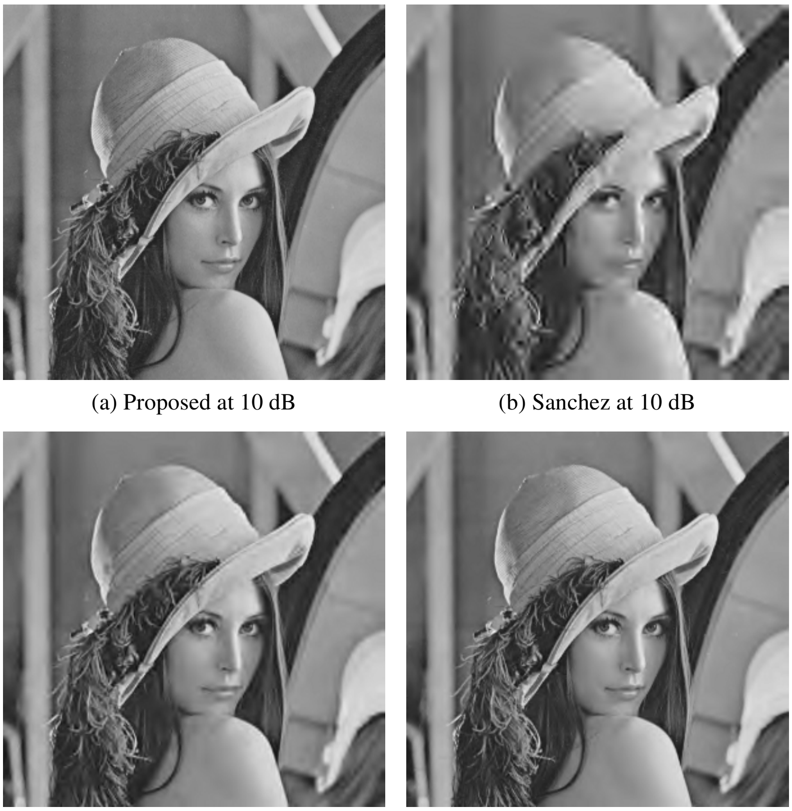


Fig. 5. Comparative performance over Rayleigh fading channel using 'Lena' image

It can be seen that the quality of the received image improves almost logarithmically with the channel E_b/N_0. At low transmission rates (≈ 0.25 bpp), our scheme achieves gain of about 1.35 dB. The gain increases to 2.5 dB at moderate transmission rates (≈ 0.5 bpp). It can be said that the gain in PSNR achieved by the proposed scheme increases with the transmission rate while maintaining superiority at lower ones. We also observe that, within this range of E_b/N_0, the new scheme provides decoding probability of 0.99~1.00 whereas Sanchez *et al.* [2] achieve about 0.80~0.98. This large variation in probability will be unpleasant for the viewers.

Comparative visual results for R_T = 4.41 bpp are presented in Fig. 6. It can be seen that the reconstructed image quality in the proposed scheme is very good or visually lossless even at low transmission power. For the same E_b/N_0, the scheme proposed by Sanchez *et al.* [2] performs poorly. At moderate E_b/N_0 (≈ 15 dB), this scheme [2] improves to good quality but some artifacts are still noticeable. At higher E_b/N_0 (≈ 20 dB), the subjective quality of both scheme are identical. Thus, the performance gap between our scheme and that of Sanchez *et al.* is quite significant at high transmission rate. However, as the transmission rate reduces, the gap reduces to certain extent. But, the new method still maintains superiority.



(a) Proposed at 10 dB

(b) Sanchez at 10 dB

(c) Sanchez at 15 dB

(d) Sanchez at 20 dB

Fig. 6. Visual results for 'Lena' image at R_T = 4.41 bpp. (a) Reconstructed after transmission via proposed scheme at E_b/N_0 = 10 dB (PSNR = 42.59 dB), (b) Sanchez *et al.* at 10 dB (PSNR = 26.88 dB), (c) Sanchez *et al.* at 15 dB (PSNR = 31.59 dB), and (d) Sanchez *et al.* at 20 dB (PSNR = 36.27 dB).

5 Conclusion

With all the performance evaluations, it can be concluded that proposed novel scheme is suitable for high-quality image transmission applications over error-prone wireless channels requiring low transmission power. The computational complexity is $O(m\Lambda)$ for each λ. In our experiments, it takes only 5 or 6 iteration to find the optimal λ. The number of iterations to find optimal solution is independent of the number of layers or channel code-rates available. The proposed algorithm has an attractive feature that enables it to optimize channel protection for a source compressed at less than, equal to or greater than available transmission rate. The novelty lies in its ability to discard the less important source layers to accommodate optimal channel protection to more important ones. However, the requirement for this scheme is that the source must have a convex rate-distortion characteristic with many layers.

Acknowledgments. The authors would like to thank Prof. D. Taubman and Dr. J. Thie from UNSW, Australia; and Faisal Zaheer from KFUPM, Saudi Arabia for valuable discussions.

References

1. JPEG2000 image coding system. ISO/IEC 15444-1/IUT-T T.800.
2. Sanchez, V., Mandal, M., K.: Efficient Channel Protection for JPEG2000 Bitstream. IEEE Trans. Circuits Syst. Video Technology. 14 (2004) 554-558
3. Banister, B., Belzer, B., Fischer, T.: Robust Image Transmission using JPEG2000 and Turbo-codes. IEEE Signal Processing Lett. 9 (2002) 117-119
4. Taubman, D., S., Marcellin, M., W.: JPEG2000 Image Compression Fundamentals, Standards and Practice. Kluwer Academic Publishers (2002)
5. Mackay, D., J., C., Neal, R., M.: Near Shannon Limit Performance of Low Density Parity Check Codes. Electron. Lett. 32 (1996) 1645-1646
6. Boulgouris, N., V., Thomos, N., Stringtzis, M., G.: Transmission of Images Over Noisy Channels Using Error-Resilient Wavelet Coding and Forward Error Correction. IEEE Trans. on Circuits and Systems for Video Technology. 13 (2003) 1170-1181
7. Shoham, Y., Gersho, A.: Efficient Bit Allocation for An Arbitrary Set of Quantizers. IEEE Trans. Accoustics, Speech and Signal Proc. 6 (1988) 1445-1453
8. Taubman, D.: High Performance Scalable Image Compression with EBCOT. IEEE Trans. Commun. 9 (2000) 1158-1170
9. Ping, L., Leung, W., K., Phamdo, N.: Low Density Parity Check Codes with Semi-Random Parity Check Matrix. Electron. Lett. 35 (1999) 38-39
10. Thie, J., Taubman, D.: Optimal Erasure Protection Assignment for Scalable Compressed Data with Small Packets and Short Channel Codewords. EURASIP Journal on Applied Signal Processing. 2 (2004) 207-219

A Novelty Detection Approach for Foreground Region Detection in Videos with Quasi-stationary Backgrounds

Alireza Tavakkoli, Mircea Nicolescu, and George Bebis

Computer Vision Lab.
Department of Computer Science and Engineering
University of Nevada, Reno, USA
{tavakkol, mircea, bebis}@cse.unr.edu

Abstract. Detecting regions of interest in video sequences is one of the most important tasks in many high level video processing applications. In this paper a novel approach based on support vector data description is presented, which detects foreground regions in videos with quasi-stationary backgrounds. The main contribution of this paper is the novelty detection approach which automatically segments video frames into background/foreground regions. By using support vector data description for each pixel, the decision boundary for the background class is modeled without the need to statistically model its probability density function. The proposed method is able to achieve very accurate foreground region detection rates even in very low contrast video sequences, and in the presence of quasi-stationary backgrounds. As opposed to many statistical background modeling approaches, the only critical parameter that needs to be adjusted in our method is the number of background training frames.

1 Introduction

In most visual surveillance systems, stationary cameras are typically used. However, because of inherent changes in the background itself, such as fluctuations in monitors and fluorescent lights, waving flags and trees, water surfaces, etc. the background of the video may not be completely stationary. In these types of backgrounds, referred to as quasi-stationary, a single background frame is not useful to detect moving regions. Pless *et al.* [1] evaluated different models for dynamic backgrounds. Typically, background models are defined independently on each pixel and depending on the complexity of the problem, use the expected pixel features (i.e. colors) [2] or consistent motion [3]. Also they may use pixel-wise information [4] or regional models of the features [5].

In [4] a single 3-D Gaussian model for each pixel in the scene is built, where the mean and covariance of the model were learned in each frame. Kalman Filtering [6] is also used to update the model. These background models were unable to represent multi-modal situations. A mixture of Gaussians modeling technique was proposed in [7] and [8] to address the multi-modality of the underlying background. There are several shortcomings for the mixture learning methods. First, the number of Gaussians needs to be specified. Second, this method does not explicitly handle spatial dependencies. Also, even with the use of incremental-EM, the parameter estimation and its convergence is noticeably slow where the Gaussians adapt to a new cluster. The convergence speed can be improved by sacrificing memory as proposed in [9], limiting its applications where mixture modeling is pixel-based and over long temporal windows. A recursive filter formulation is proposed by Lee in [10]. However, the problem of specifying the number of Gaussians as well as the adaptation in later stages still exists. Also this model does not account for the situations where the number of Gaussians changes due to occlusion or uncovered parts of the background.

In [2], El Gammal *et al.* proposed a non-parametric kernel density estimation method (KDE) for pixel-wise background modeling without making any assumption on its probability distribution. Therefore, this method can easily deal with multi-modality in background pixel distributions without determining the number of modes in the background. However, there are several issues to be addressed using non-parametric kernel density estimation. First, these methods are memory and time consuming. For each pixel in each frame the system has to compute the average of all the kernels centered at each training feature vector. Second, the size of the temporal window used as the background buffer needs to be specified. Too small a window increases the estimation speed, while it does not incorporate enough history for the pixel, resulting in a less accurate model. Also the adaptation will be problematic by using small window sizes. Increasing the window size improves the accuracy of the model but with the cost of more memory requirements and slower convergence. Finally, the non-parametric KDE methods are pixel-wise techniques and do not use the spatial correlation of the pixel features. In order to adapt the model a sliding window is used in [11]. However the model convergence is problematic in situations where the illumination suddenly changes.

In order to update the background for scene changes such as moved objects, parked vehicles or opened/closed doors, Kim *et al.* in [12] proposed a layered modeling technique. This technique needs an additional model called *cache* and assumes that the background modeling is performed over a long period. It should be used as a post-processing stage after the background is modeled.

In methods that explicitly model the background density estimation, foreground detection is performed by comparing the estimated probabilities of each pixel with a global threshold [2], or local thresholds [13]. Also there are several parameters that need to be estimated from the data to achieve an accurate density estimation for background. In [11] a binary classification technique is used to detect foreground regions by a maximum likelihood method. Since in these techniques the probability density function of the background is estimated, the model accuracy is bounded to the accuracy of the estimated probability.

In this paper a novel approach is proposed to label pixels in video sequences into foreground and background classes using support vector data description [14]. As opposed to parametric and non-parametric density estimation techniques, in this method the model is not the probability function of the background or foreground.

It can be considered as analytical description of the decision boundary between background and foreground classes. This modeling technique addresses several issues in the traditional density estimation approaches.

First, the model accuracy is not bounded to the accuracy of the estimated probability density functions. Second, the memory requirements of the proposed technique are less than those of non-parametric techniques. In non-parametric density estimation methods, pixel feature vectors for all background training frames need to be stored to regenerate the probability of pixels in new frames. It can be problematic for large frames sizes and temporal windows. In our technique, in order to classify new pixels, they are compared only with the support vectors, which in practice are much fewer than the actual number of frames in the temporal window. Third, because support vector data description explicitly models the decision boundary of the known class, it can be used for novelty detection and single class-classification without a need to threshold any values. This results in less parameter tuning and automatic classification. Finally, the performance of the classifier in terms of false positive and false negatives can be controlled from within the framework. The proposed method is a novel approach that explicitly addresses the one-class classification problem, since in foreground region detection we do not have samples of foreground regions in the training steps of the system. This issue, has not been addressed in any of the traditional techniques.

The rest of this paper is organized as follows. In Section 2 a brief review of the support vector data description is presented. In Section 3, the proposed method for foreground region detection is discussed. Section 4 shows experimental results of our method on synthetic and real-world data, and the performance of classifier is compared with the existing techniques. Finally, conclusions of the proposed method are drawn in Section 5 and future extensions to this work are discussed.

2 Support Vector Data Description

Data domain description concerns the characteristics of a data set [14]. The boundary of the dataset can be used to detect novel data or outliers. A normal data description gives a closed boundary around the data. The simplest boundary can be represented by a hyper-sphere. The volume of this hyper-sphere with center a and radius R should be minimized while containing all the training samples x_i. To allow the possibility of outliers in the training set, slack variables $\epsilon_i \geq 0$ are introduced. The error function to be minimized is defined as:

$$F(R, a) = R^2 + C \sum_i \epsilon_i \tag{1}$$

subject to the constraints:

$$\|x_i - a\|^2 \leq R^2 + \epsilon_i \quad \forall i \tag{2}$$

In equation (1), C is a trade-off between simplicity of the system and its error. We call this parameter confidence parameter. After incorporating the constraints (2) into the error function (1) by Lagrange multipliers we have:

$$L(R, a, \alpha_i, \gamma_i, \epsilon_i) = R^2 + C\sum_i \epsilon_i - \sum_i \alpha_i \left[R^2 + \epsilon_i - \left(\|x_i - a\|^2\right)\right] - \sum_i \gamma_i \epsilon_i \quad (3)$$

L should be maximized with respect to Lagrange multipliers $\alpha_i \geq 0$ and $\gamma_i \geq 0$ and minimized with respect to R, a and ϵ_i. Lagrange multipliers γ_i can be removed if the constraint $0 \leq \alpha_i \leq C$ is imposed. After solving the optimization problem we have:

$$L = \sum_i \alpha_i (x_i \cdot x_i) - \sum_{i,j} \alpha_i \alpha_j (x_i \cdot x_j) \quad \forall \alpha_i : 0 \leq \alpha_i \leq C \quad (4)$$

When a new sample satisfies the inequality in (2), then its corresponding Lagrange multipliers are $\alpha_i \geq 0$, otherwise they are zero. Therefore we have:

$$\begin{aligned} &\|x_i - a\|^2 < R^2 \to \alpha_i = 0, \gamma_i = 0 \\ &\|x_i - a\|^2 = R^2 \to 0 < \alpha_i < C, \gamma_i = 0 \\ &\|x_i - a\|^2 > R^2 \to \alpha_i = C, \gamma_i > 0 \end{aligned} \quad (5)$$

From the above, we can see that only samples with non-zero α_i are needed in the description of the data set, therefore they are called *support vectors* of the description. To test a new sample y, its distance to the center of the hyper-sphere is calculated and tested against R.

In order to have a flexible data description as opposed to the simple hyper-sphere discussed above a kernel function $K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$ is introduced. This maps the data into a higher dimensional space, where it is described by the simple hyper-sphere boundary.

3 The Proposed Method

The methodology described in section 2 is used in our technique to build a descriptive boundary for each pixel in the background training frames to generate its model for the background. Then these boundaries are used to classify their corresponding pixels in new frames as background and novel (foreground) pixels. There are several advantages in using the Support Vector Data Description (SVDD) method in detecting foreground regions:

- The proposed method, as opposed to existing statistical modeling methods, explicitly addresses the single-class classification problem. Existing statistical approaches try to estimate the probability of a pixel being background, and then use a threshold for the probability to classify it into background or foreground regions. The disadvantage of these approaches is in the fact that it is impossible to have an estimate of the foreground probabilities, since there are no foreground samples in the training frames.

```
1. Initialization
     1.1. Confidence parameter: $C$
     1.2. Number of training samples: Trn_No
     1.3. Kernel bandwidth $\sigma$
2. For each training frame
     2.1. For each pixel $\mathbf{x}_{ij}$
           2.1.1. OC(i,j)$\leftarrow$ Train($\mathbf{x}_{ij}$[1 :Trn_No])
3. For new frames
    3.1. For each pixel $\mathbf{x}_{ij}$
          3.1.1. Desc(i,j)$\leftarrow$ Test($\mathbf{x}_{ij}$[Current Frame],OC(i,j))
          3.1.2. Label pixel as foreground or background
                 based on Desc(i,j).
```

Fig. 1. The proposed algorithm

- The proposed method has less memory requirements compared to non-parametric density estimation techniques, where all the training samples for the background need to be stored in order to estimate the probability of each pixel in new frames. The proposed technique only requires a very small portion of the training samples, *support vectors*, to classify new pixels.
- The accuracy of our method is not limited to the accuracy of the estimated probability density functions for each pixel. Also the fact that there is no need to assume any parametric form for the underlying probability density of pixels gives the proposed method superiority over the parametric density estimation techniques, i.e. mixture of Gaussians.
- The efficiency of our method can be explicitly measured in terms of false reject rates. The proposed method considers a goal for false positive rates, and generates the description of data by fixing the false positive tolerance of the system. This helps in building a robust and accurate background model.

Figure 1 shows the proposed algorithm in pseudo-code format[1]. The only critical parameter is the number of training frames (`Trn_No`) that needs to be initialized. The support vector data description confidence parameter C is the target false reject rate of the system. This is not a critical parameter and accounts for the system tolerance. Finally the Gaussian kernel bandwidth, σ does not have a particular effect on the detection rate as long as it is not set to be less than one, since features used in our method are normalized pixel chrominance values. For all of our experiments we set $C = 0.1$ and $\sigma = 5$.

In the first step of the algorithm, for each pixel in the scene a single class classifier is trained by using its values in the background training frames. This classifier consists of the description boundary and support vectors, as well as a threshold used to describe the data. In the next step, each pixel in the new frames is classified as background or foreground using its value and its corresponding

[1] The proposed method is implemented in MATLAB 6.5, using Data Description toolbox [15].

classifier from the training stage. Details of training of each classifier and testing for new data samples are provided in section 2.

Feature vectors x_{ij} used in the current implementation are $x_{ij} = [cr, cg]$, where cr and cg are the red and green chrominance values for pixel (i, j). Since there is no assumption on the dependency between features, any feature value such as vertical and horizontal motion vectors, pixel intensity, etc. can be used.

4 Experimental Results

In this section, the experimental results of the proposed method are presented on both synthetic and real data. The experiments are conducted to compare the results of support vector data description in classification problems with those of traditional methods, such as mixture of Gaussians (MoG), Kernel Density Estimation (KDE) and k-nearest neighbors (KNN).

4.1 Synthetic Data Sets

In this section we use a synthetic data set, which represents randomly distributed training samples with an unknown distribution function (*banana* data set). Figure 2 shows a comparison between different classifiers. This experiment is performed on 150 training samples using support vector data description (SVDD), mixture of Gaussians (MoG), kernel density estimation (KDE) and k-nearest neighbors (KNN).

Parameters of these classifiers are manually determined to give a good performance. For all classifiers the confidence parameter is set to be 0.1. In MoG, we used 3 Gaussians. Gaussian kernel bandwidth in the KDE classifier is considered $\sigma = 1$, for the KNN we used 5 nearest neighbors, and for the SVDD classifier the Gaussian kernel bandwidth is chosen to be 5.

Figure 2(a) shows the decision boundaries of different classifiers on 150 training samples from *banana* dataset. As it can be seen from Figure 2(b), SVDD

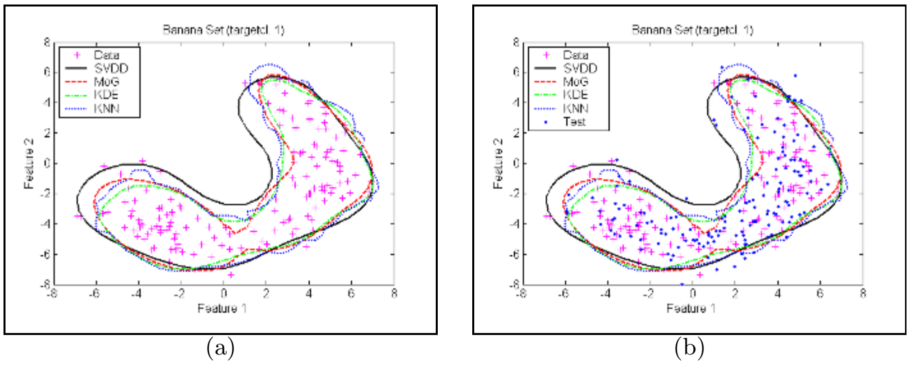


Fig. 2. Comparison between different classifiers on a synthetic data set: (a)- Decision boundaries of different classifiers after training. (b)- Data points (blue dots) outside decision boundaries are false rejects.

generalizes better than the other three classifiers and classifies the test data more accurately. In this Figure the test data is composed of 150 samples drawn from the same probability distribution function as the training data, thus should be classified as the known class.

Table 1. Comparison of False Reject Rate and Recall Rate for different classifiers

Method	SVDD	MoG	KDE	KNN
FRR	0.1067	0.1400	0.1667	0.1333
RR	0.8933	0.8600	0.8333	0.8667

Here we need to define the False Reject Rate (FRR) and Recall Rate (RR) for a quantitative evaluation. By definition, FRR is the percentage of missed targets, and RR is the percentage of correct prediction (True Positive rate). These quantities are given by:

$$\mathrm{FRR} = \frac{\#\text{Missed targets}}{\#\text{Samples}} \qquad \mathrm{RR} = \frac{\#\text{Correct predictions}}{\#\text{Samples}} \tag{6}$$

Table 1 shows a quantitative comparison between different classifiers. In this table, FRR and RR of classifiers are compared after training them on 150 data points drawn from an arbitrary probability function and tested on the same number of samples drawn from the same distribution. As it can be seen from the above example, the FRR for SVDD classifier is less than that of the other three, while its RR is higher. This proves the superiority of this classifier in case of single class classification over the other three techniques.

Table 2. Need for manual optimization and number of parameters

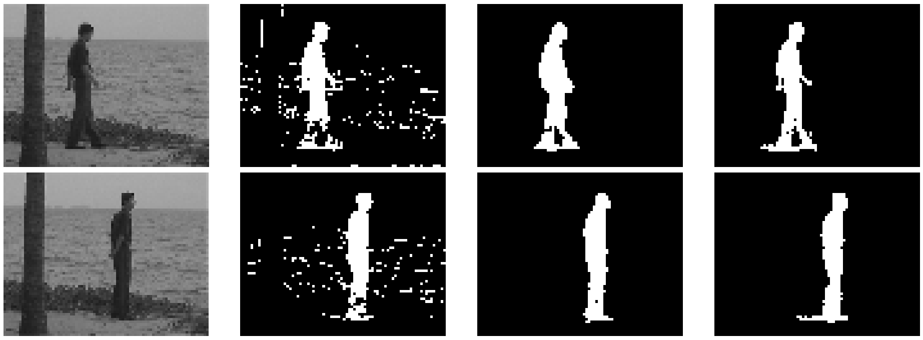
Method	SVDD	MoG	KDE	KNN
No. of parameters	1	4	2	2
Need manual selection	No	Yes	Yes	Yes

Table 2 compares the number of parameters that each classifier has and the need for manually selecting these parameters for a single class classification problem. As it can be seen, the only method that automatically determines data description is the SVDD technique. In all of the other classification techniques there is at least one parameter that needs to be manually chosen to give a good performance.

Table 3 shows memory requirements for each classifier. As it can be seen from the table, SVDD requires much less memory than the KNN and KDE methods, since in SVDD we do not need to store all the training data. Only the MoG method needs less memory than the SVDD technique, but in MoG methods we need to manually determine the number of Gaussians to be used which is not practical when we are training one classifier per pixel in real video sequences.

Table 3. Comparison of memory requirements for different classifiers

Method	SVDD	MoG	KDE	KNN
Memory needs (bytes)	1064	384	4824	4840



(a) Water surface (b) MoG (c) KDE (d) SVDD

Fig. 3. Water surface video: comparison of methods in presence of irregular motion

4.2 Real Videos

In this section, foreground detection results of our method on real video sequences are shown and compared with the traditional statistical modeling techniques.

Comparison in the Presence of Irregular Motion. By using the *water surface* video sequence, we compare the results of foreground region detection using our proposed method with a typical KDE [13] and MoG [7]. For this comparison the sliding window of size L=150 is used in the KDE method. The results of MoG are shown in Figure 3(b), the KDE method results are shown in Figure 3(c) and the foreground masks detected by the proposed technique are shown in Figure 3(d). As it can be seen, the proposed method gives better detection since it generates a more accurate descriptive boundary on the training data, and does not need a threshold to classify pixels as background or foreground.

Comparison in Case of Low Contrast Videos. Figure 4 shows the result of foreground detection using the proposed method in the *hand shake* video sequence and compares this result with that of the KDE method. As it can be seen from Figure 4(b) and 4(c), the proposed method achieves better detection rates compared to the KDE technique. Notice that in the KDE technique presented in the figure, the system tries to find the best parameters for the classifier in order to achieve its best performance.

Detection Results in Difficult Scenarios. Figure 5 shows results of the proposed foreground detection algorithm in very difficult situations. In Figure 5(a) and 5(b) the irregular motion of water in the background make it difficult to detect true foreground regions. In Figure 5(c) there are flickers in the lighting.

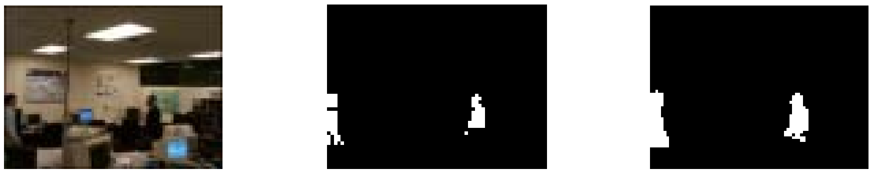

(a) Hand shake sequence (b) KDE (c) SVDD

Fig. 4. Hand shake video: comparison of methods in case of low contrast videos

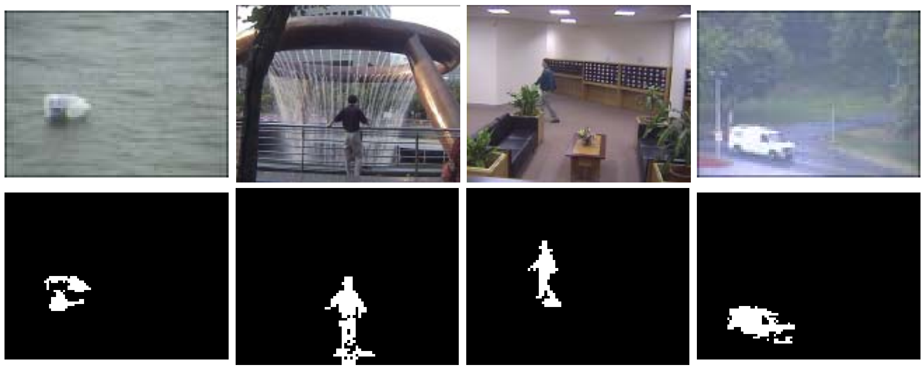

(a) Water (b) Fountain (c) Lobby (d) Water surface

Fig. 5. Foreground detection results

Our system accurately detects the foreground regions in all of these situations. Also the car in Figure 5(d) is detected accurately by our method despite the presence of waving tree branches and the rain in the background.

5 Conclusions and Future Work

In this paper a novel approach is proposed to label pixels in video sequences into foreground and background classes using support vector data description. The contributions of our method can be described along the following directions:

– The model accuracy is not bounded to the accuracy of the estimated probability density functions.
– The memory requirement of the proposed technique is less than that of non-parametric techniques.
– Because support vector data description explicitly models the decision boundary of the known class, it is suitable for novelty detection without the need to use thresholds. This results in less parameter tuning.
– The classifier performance in terms of false positive is controlled explicitly.

One direction of future investigation is to use this work in non-parametric tracking approaches. Also we are investigating the effect of adaptive kernel bandwidth parameters on the performance of the system.

Acknowledgement

This work was supported by the NSF-EPSCoR Ring True III award EPS0447416 and by the Office of Naval Research award N00014-06-1-0611.

References

1. Pless, R., Larson, J., Siebers, S., Westover, B.: Evaluation of local models of synamic backgrounds. Proceedings of the CVPR **2** (2003) 73–78.
2. Elgammal, A., Duraiswami, R., Harwood, D., Davis, L.: Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. Proceedings of the IEEE **90** (2002) 1151–1163.
3. Pless, R., Brodsky, T., Aloimonos, Y.: Detecting independent motion: The statistics of temporal continuity. IEEE Transactions on PAMI **22** (2000) 68–73.
4. Wern, C., Azarbayejani, A., Darrel, T., Petland, A.: Pfinder: real-time tracking of human body. IEEE Transactions on PAMI **19** (1997) 780–785.
5. Toyama, K., Krumm, J., Brumitt, B., Meyers, B.: Wallflower: principels and practive of background maintenance. Proceedings of ICCV **1** (1999) 255–261.
6. Koller, D., adn T. Huang, J.W., Malik, J., Ogasawara, G., Rao, B., Russel, S.: Towards robust automatic traffic scene analysis in real-time. ICPR **1** (1994) 126–131.
7. Stauffer, C., Grimson, W.: Learning patterns of activity using real-time tracking. IEEE Transactions on PAMI **22** (2000) 747–757.
8. Friedman, N., Russell, S.: Image segmentation in video sequences: A probabilistic approach. Annual Conference on Uncertainty in Artificial Intelligence (1997) 175–181.
9. McKenna, S., Raja, Y., Gong, S.: Object tracking using adaptive color mixture models. Proceedings of Asian Conferenc on Computer Vision **1** (1998) 615–622.
10. Lee, D.S.: Effective gaussian mixture learning for video background subtraction. IEEE Transactions on PAMI **27** (2005) 827–832.
11. Mittal, A., Paragios, N.: Motion-based background subtraction using adaptive kernel density estimation. Proceedings of CVPR **2** (2004) 302–309.
12. Kim, K., Harwood, D., Davis, L.S.: Background updating for visual surveillance. Proceedings of the International Symposium on Visual Computing **1** (2005) 337–346.
13. Tavakkoli, A., Nicolescu, M., Bebis, G.: Automatic robust background modeling using multivariate non-parametric kernel density estimation for visual surveillance. Proceedings of the International Symposium on Visual Computing **LNSC 3804** (2005) 363–370.
14. Tax, D.M.J., Duin, R.P.: Support vector data description. Machine Learning **54** (2004) 45–66.
15. Tax, D.: Ddtools, the data description toolbox for matlab (2005) version 1.11.

Procedural Image Processing for Visualization

Xiaoru Yuan and Baoquan Chen

Department of Computer Science and Engineering
University of Minnesota at Twin Cities, MN 55455, USA
{xyuan, baoquan}@cs.umn.edu

Abstract. We present a novel Procedural Image Processing (PIP) method and demonstrate its applications in visualization. PIP modulates the sampling positions of a conventional image processing kernel (e.g. edge detection filter) through a procedural perturbation function. When properly designed, PIP can produce a variety of styles for edge depiction, varying on width, solidity, and pattern, etc. In addition to producing artistic stylization, in this paper we demonstrate that PIP can be employed to achieve various visualization tasks, such as contour enhancement, focus+context visualization, importance driven visualization and uncertainty visualization.

PIP produces unique effects that often either cannot be easily achieved through conventional filters or would require multiple pass filtering. PIP perturbation functions are either defined by analytical expressions or encoded in pre-generated images. We leverage the programmable fragment shader of the current graphics hardware for achieving the operations in real-time.

1 Introduction

A key objective of visualization is to effectively deliver information by emphasizing important features while hiding unimportant ones. Image processing methods have been employed as a viable tool for illustration[1] where geometric features such as silhouettes, ridges, and valleys are extracted through image processing (e.g. edge detection) on intermediately generated geometric images. However, while existing image processing algorithms are effective in detecting features, they fall short in stylizing their appearance.

In this paper, we propose a new image filtering operation termed procedural image processing (PIP) that can stylize features *during* the process of detecting them. While a conventional image filter is defined by a kernel matrix with a fixed sampling and weighting pattern, a PIP filter has its sampling positions modulated by a perturbation function. When the perturbation function is designed properly, the detected features can be either enhanced or deprecated to display certain stylization. PIP produces unique effects that often either cannot be easily achieved through conventional filters or would require multiple pass filtering. The PIP filter has been applied to visualize isosurface in volume [2], in this paper, we extend its application in 2D image, 3D geometry visualization and 3D volume visualization. We discuss several perturbation functions or patterns and demonstrate their effectiveness on various visualization tasks, such as contour enhancement, focus+context visualization, and uncertainty or importance driven visualization.

The most recent advances in graphics hardware provide the ability to interactively perform image processing on GPU [3]. PIP operation adds little overhead to a conventional image filtering operation. When perturbation patterns encoded in a texture are passed to GPU with a few additional parameters, the added computation for PIP implementation is a few texture lookup operations and texture coordinates calculation. All PIP-based renderings are performed wholly on hardware through fragment programming and real-time performance is achieved.

2 Related Work

Many visualization systems have been striving to achieve improved effectiveness by selectively depicting important data features. For example, to visualize highly complex volume data, gradient operations have been employed to enhance boundaries via opacity modulation [4]. The more recent volume illustration systems have integrated gradient modulation into volume stylization [5]. Gradient information has also been used in high-dimensional transfer function design [6,7].

Image processing operators, including extracting edges, have been employed to achieve non-photorealistic rendering (NPR) of 3D objects. Image processing is performed on intermediately generated geometric images (G-buffer) to depict geometric features such as silhouettes, ridges, and valleys [8,1]. As only 2D processing in image space is required, image-based NPR methods can be more efficient than traditional object-based NPR methods which perform geometric feature extraction operations on 3D geometry.

Generally, an image processing filter defines a weighting function that is usually discretized and stored in a matrix, often called filter kernel. To process a pixel, the filter kernel is centered at the pixel, neighboring pixels are sampled based on the matrix grids and multiplied with matrix values (weights) and are finally summed together. Traditionally a linear edge filter is fixed in terms of both its weights and sampling pattern (matrix grid). Recently, a new filter called bilateral filter [9] has been proposed in which the weight of each filter sample is changed based on the difference between its value and the averaged value of all samples within the filter's footprint. Nevertheless, the output of a conventional filter is in the form of pixelized lines that lack solidity and styles. Since images are processed pixel by pixel and edge information is available only after the entire image processing is done, stylizing pixelized features such as halftoning [10], or overdrawing parameterized strokes on them usually needs additional pass(es) of processing.

In the remainder of the paper, we first introduce the basic elements of Procedural Image Processing (PIP) (Section 3), then describe its various applications(Section 4).

3 Procedural Image Processing (PIP)

An image filter kernel can be represented by a matrix mask W. Let the input image F be a pixel array $f(v)$, where v is the vector representing pixel position (x,y); W denotes a $m \times n$ filter kernel applied to the input image and the output image G has a pixel array $g(v)$. The output image can be computed by the expression:

$$g(v) = F \bigotimes W = \sum_{s=-a}^{s=a} \sum_{t=-b}^{t=b} w(r) f(v+r), \tag{1}$$

where $a = (m-1)/2$, $b = (n-1)/2$ and r denotes a filter element position (s,t). In cases presented in this paper, $m = n = 3$ (i.e., 9 samples for each filter). The filtering convolution of $F \bigotimes M$ at pixel $f(v)$ is obtained as usual by centering the filter matrix at the pixel, multiplying the matrix elements with their corresponding pixels and summing the results. Figure 2(a) illustrates an isosurface rendering image after conventional Sobel edge filtering. Silhouette lines have a one-pixel width.

For PIP filtering, a filter's sampling positions are perturbed by a procedural function $P(v,r)$, which is defined in the entire image domain:

$$g_{pip}(v) = \sum_{s=-a}^{s=a} \sum_{t=-b}^{t=b} w(r) f(v + P(v,r)). \tag{2}$$

Let us draw upon some intuitions here. If $P(v,r) = r$, the equation 2 is equivalent to the equation 1 and the filter becomes a regular filter. When the amplitudes of perturbation function $P(v,r)$ increases, sampling positions are moved away from the center pixel v, and pixels further away from the filtering center will be sampled. This can be considered as enlarging the filter kernel size (but the number of samples is kept the same). Therefore, slow pixel-value variations may be amplified, resulting in thicker edges. To gain more flexible control over the perturbation functions, we decompose a perturbation function into two components: scaling and translation. The resulting perturbation function is then expressed as:

$$P(v,r) = P_{scale}(v) r + P_{trans}(v). \tag{3}$$

The scaling factor is uniform for every direction. P_{trans} represents the translation vector. The perturbation functions can be encoded in textures for GPU implementation.

The effects of applying PIP with scaling and translation perturbations are demonstrated in Figure 1 for edge detection on 1D signals. Figure 1(a) is the original 1D signal with an edge. Figure 1(b) is the output of Figure 1(a) applied with a regular edge detection filter. The edge filter is a 1D matrix $|-1,0,1|$, each sampling position is exactly on the pixel grid. In Figure 1(c), a PIP with $P_{scale} = 1.5$ is introduced. The result edge is wider. We will demonstrated and discuss such edge widening effect in the next section. In Figure 1(d), a translation with offset of 0.5pixel in negative axial direction is applied. The resulting edge is shifting to positive axial direction correspondingly. In above cases, the same PIP perturbation function is applied throughout whole input signal. When applying different PIP perturbation functions in different regions, stylization variation can be introduced to the final rendering. By providing careful user control on such perturbation functions, various visualization effects can be generated.

Figures 2(b) and (c) demonstrate the results after applying only scaling perturbation. Larg constant scaling values are used throughout the entire respective image. Thicker edges than those from regular Soble filter (Figure 2(a)) are obtained. Such edge thickening cannot be achieved by simply decreasing the threshold, as that will introduce unwanted noises (Figure 2(d)).

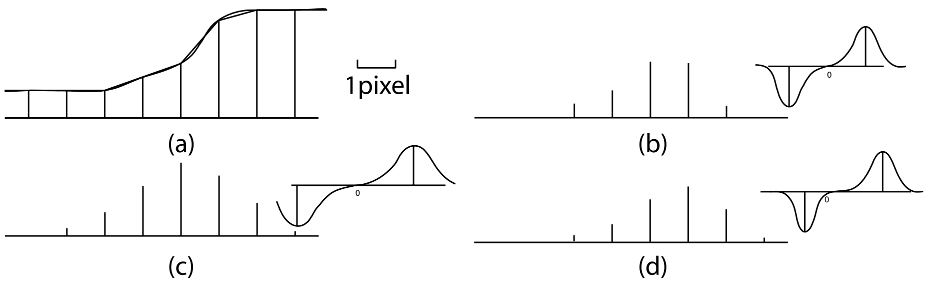


Fig. 1. 1D Procedural Image Processing (Edge Detection). (a) input signal, (b) signal output of a normal edge detection filter, (c) PIP filter ($P_{scale} = 1.5$), (d) PIP filter, shift a normal edge detection filter 0.5 pixel leftwards, ($P_{trans} = -0.5$).

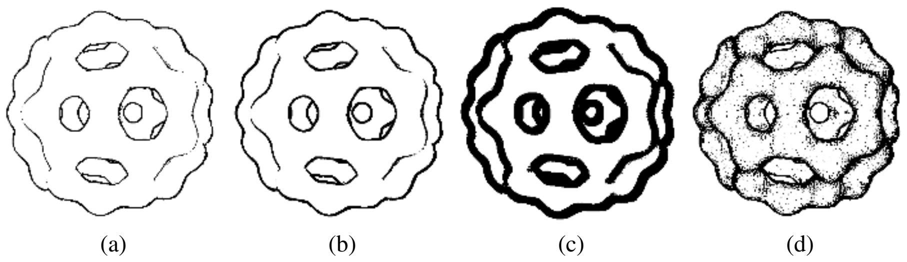


Fig. 2. 2D edge detection of isosurface (the isosurface is extracted from isosurface is extracted from Bucky ball data): (a) regular filter ($P_{scale} = 1.0$), threshold$= 0.65$; (b) PIP filter ($P_{scale} = 2.0$), threshold$= 0.65$; (c) PIP filter ($P_{scale} = 8.0$), threshold$= 0.65$; (d) PIP filter ($P_{scale} = 1.0$), threshold$= 0.05$

We must point out that the net behavior of the PIP filter also depends on the underlying image contents. The goal of the perturbation function design is to make the resulting feature illustration appearing random at small scales while conforming with large-scale stylizations driven by large-scale perturbation patterns.

The perturbation function can be normalized to the range of $[0, 1]$. Then at the processing time, the looked-up perturbation value needs to be multiplied with a constant scaling factor. Since the perturbation functions are usually periodic, we simply need to store one cycle of the function, which can be encoded in a much smaller texture image. During the processing, the texture image is tiled together to cover the entire image domain. Translation perturbation can be encoded in similar forms.

4 Applications

We demonstrate the capability of PIP in both image processing and visualization.

4.1 2D Image Processing

Various filtering results can be obtained by applying PIP to 2D images. Figure 3 shows several different PIP operations on the Lena image using different perturbation functions.

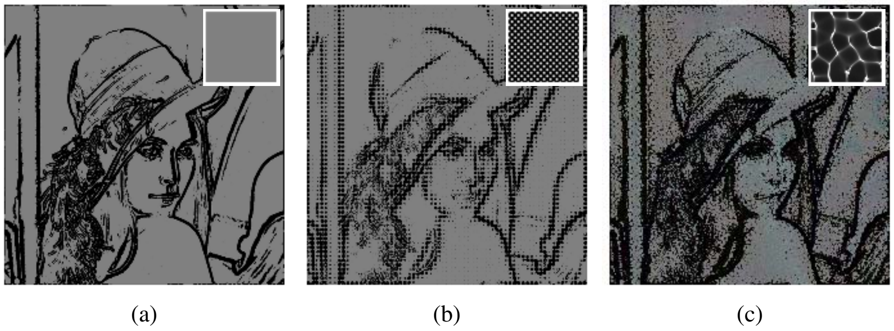


Fig. 3. PIP processed results of the Lena image. For each image, the smaller image in the white frame encodes scaling perturbation function.

Only the scaling perturbation is applied in all the examples here. For each image, the smaller image in the top right encodes the scaling perturbation function. In Figure 3(a), a constant scaling factor of 4 is used. Thickened edge lines are obtained. In Figure 3(b), a regular dot pattern is used as the scaling perturbation. The processed edges demonstrate a dot pattern, resembling a halftoning effect. In Figure 3(c), a swirl-like perturbation texture is used; the swirling perturbation patterns are carried over to edge depiction, resembling pen-and-ink drawing. In Figure 3(d), a water caustics texture is used as perturbation function. The resulting image displays beehive patterns resembling cracked painting.

4.2 Geometric Data Visualization

We demonstrate here different visualization effects that PIP can generate for 3D geometric models. The 3D rendered geometric buffers, like those used in image-based NPR [1], can encode additional information such as depth and normal. We utilize this additional information to modulate PIP operation to obtain additional control over data visualization.

In the first example we use depth values to control the PIP operation for modulating edge appearance based on depth, demonstrating a different kind of 'depth cueing'. Figure 4(a) shows such a result. The scaling perturbation function is defined as a constant (encoded in the left grey image in the bottom row). The depth image is on the right in the bottom row, while the regular color image is in the middle and is the one on which the PIP processing operates. Even though the scaling perturbation function is specified as a constant, it is inversely scaled by the corresponding depth value at each pixel. Thus more distant objects get a smaller scaling perturbation, resulting in a thinner detected edge.

In the next example we demonstrate that a different image portion or different part of a 3D object can use a different perturbation function to demonstrate different styles across the image. This style difference can be used to emphasize different aspects or convey certainty/uncertainty in data visualization. In our experiment, we define an importance map with values between $[0,1]$ (1: most important; 0: least important). We

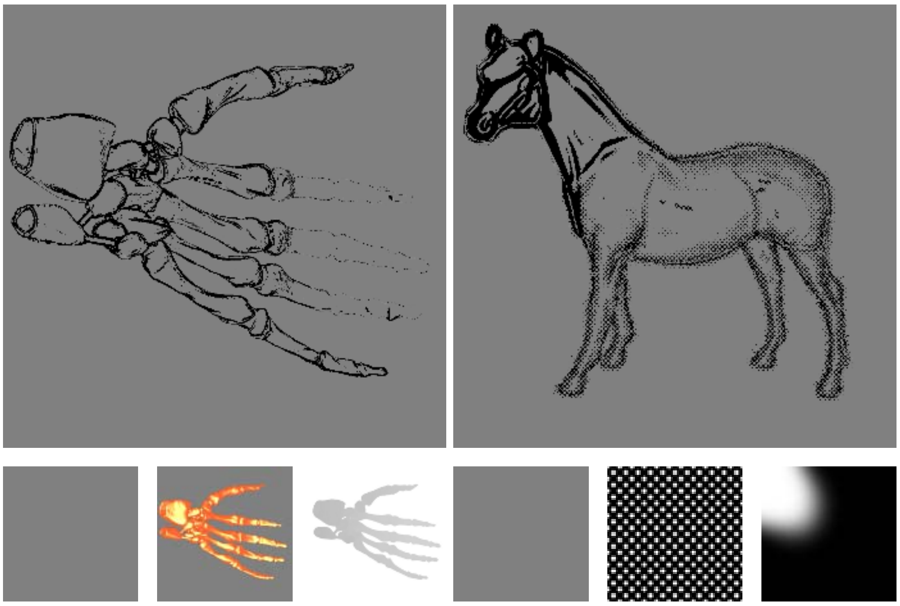


Fig. 4. Various effects generated by PIP: (a) depth cueing and (b) attention depiction. The bottom row of (a) shows (from left to right) the scaling perturbation pattern, the regular color image, the depth image. The bottom tow of (b) shows (from left to right) the two scaling perturbation functions for the most and least importance values and the importance map (bright pixel indicates higher importance).

then define two separate scaling perturbation functions associated with 0 and 1 importance value. For an importance value between 0 and 1, the perturbation function is interpolated between the two pre-defined ones. In Figure 4(b), the first two images in the bottom row illustrate the two pre-defined perturbation patterns; the uniform grey pattern is used for the most important value as it tends to strengthen edges, while the dot pattern will do the opposite. The rightmost image in the bottom row is the importance map. The top left corner of this image (corresponding to the horse head) is specified as the attention region (with high importance values). The resulting image shown on the top conforms with the design – the head is illustrated using thick dark lines and the rest of the body is depicted using small dots. Although in this example the importance map is specified in the image domain, it can be generated dynamically in 3D rendering. For example, when an attention point is specified on a 3D object, its projection on the screen can be the attention center. The attention center can also be obtained through eye or gaze tracking.

4.3 Volume Visualization

We also apply PIP to volume visualization. PIP operations on both isosurface rendering and direct volume rendering are demonstrated in the following examples.

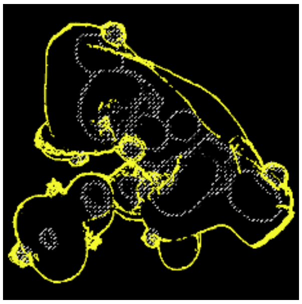


Fig. 5. PIP enhanced silhouettes of two isosurfaces

Isosurface Rendering: The isosurface method renders the surface that is defined by an isovalue in the volume. Same as for 3D polygon models, PIP generates silhouette and contour edges from the intermediate geometric buffers of the isosurface(s). Figure 5 shows two layers of outlines generated by PIP. Silhouettes of the inner isosurface are depicted by dashed lines The rendered silhouettes represent important geometry features of the isosurface; the variation of stylization can be used to depict other associated data information, such as uncertainty.

Direct Volume Rendering: Throughout the examples that we discuss here, we perform gradient modulated volume rendering. The gradient is computed using 3D central difference. The computed gradient magnitudes further modulate the opacity of the sample. Figure 6(a) and (b) show a conventional volume rendering of the Neghip data, without and with gradient modulation, respectively.

Next we illustrate a variety of interesting visualizing effects that can be generated by applying PIP at different stages of different purposed volume rendering. The first example illustrates how PIP can be used for importance driven or focus emphasis visualization. Research has been done to utilize different rendering methods such as direct volume rendering, maximum intensity rendering and NPR rendering to differentiate focus and context regions [11,12]. Here we seek to design perturbation functions in PIP to achieve the same goal. We define a scalar value d to represent the degree of interest. In the region of interest (ROI), $d = 1.0$, otherwise $d < 1.0$. We then use this d value as the scaling factor in PIP kernel. Features (surface boundaries) are indeed much enhanced in the ROI when this PIP is applied to gradient modulated volume rendering (like in Figure 6(b)), while becoming less visible in regions outside the focus center. Figure 6(c) illustrates the result.

The fourth example shows application of PIP to uncertainty visualization [13,14]. To simulate an uncertainty distribution, we assume the data at the ROI is the most accurate, but uncertainty increases when moving away from the ROI. There are several possible ways of achieving this. One intuitive way is to generate a noise function (ranging from 0 to 1) and then directly modulate opacity with this noise value.

We now describe our approaches of using PIP for achieving uncertainty visualization. First, we define a noise function, which is multiplied by $1-d$ and added with d and is then used as the the scaling function of PIP. Hence, the ROI receives no noise

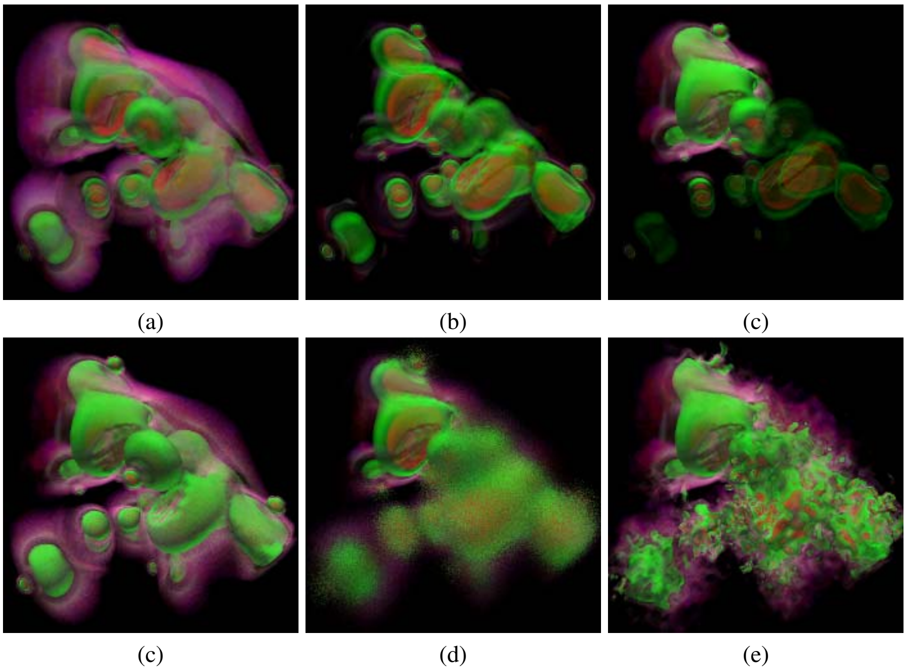


Fig. 6. Various effects of volume rendering: (a) regular volume rendering; (b) gradient enhanced volume rendering (regular filter); (c) importance driven visualization; (d-f) uncertainty visualization ((d) opacity directly modulated by noise, (e) noise as the translation function in PIP, (f) similar to (e), but with lower noise frequency)

while the noise values increase as one moves away from the focus center. The consequence is that the thickness of the boundary is randomly varied (hence appearing rough) in the uncertainty region. Figure 6(d) illustrates this effect. Our method to depict uncertainty is to define a translation function of PIP. The translation direction is constant, but the amount of translation is determined by $(1-d)*noise$. Figure 6(e) shows the resulting effect in which the surface boundaries become cloudy at the uncertain region. In Figure 6(f), we apply a lower frequency noise than that is used in Figure 6(e). In this image, surface boundaries can be seen as being randomly deformed. Through the above examples, PIP provides another set of opportunities for visualizing data uncertainty, especially when surfaces boundaries are to be depicted.

5 Implementation Notes, Discussions and Conclusion

All our experiments have been performed on a Dell Precision 530 workstation with a 256MB GeForce FX5900 Ultra graphics card. We use the newly introduced Cg language [15] for hardware fragment programming. Specifically, NV_fragment_program (fp30 profile in Cg program) OpenGL extension [3] is used in our implementation. All the operations involved in PIP are implemented in hardware. Perturbation functions are encoded in texture images. Texture lookup, sample transformation and interpolation, and filter convolution are some typical sub-operations in the implementation of the PIP operation. The PIP implementation can be fully integrated with hardware assisted volume rendering, which leverages 3D texture mapping hardware [16]. The PIP filter operation is directly performed in the fragment program where 3D texture lookups are performed. The overhead of PIP operation is almost negligible. Because of the hardware implementation, all rendering demonstrated in the paper is real time. The rendering times for the stylized Lena images in Figures 3 are around 0.15 ms. The 2D image rendering time is close to the PIP processing time which is much smaller comparing with model rendering time of mesh or volume. The rendering time of the horse model (97K triangles) in Figure 4 is 11.2 ms and the hand model (655K triangles) is 68 ms. We didn't observe obvious performance difference when the PIP operation is turned on.

Although PIP represents a simple extension of conventional image processing, a variety of effects can be achieved *during* image filtering without resorting to any pre- or post-processing. The stylization is achieved by perturbation patterns which modulate filters' sampling positions. The effects are unique that often either cannot be achieved through conventional filters or would require multiple pass filtering. Our method works on both 2D images and 3D data and can achieve a wide range of visualization tasks, such as depth cueing, focus+context visualization, importance driven visualization, and uncertainty visualization.

We have identified two immediate avenues for future work. First, we plan to investigate more perturbation functions in hope of achieving additional effects. Along this line, we seek to understand the relationship between perturbation functions and their resulting effects in greater depth. Second, although we have concentrated on edge detection filter so far, we aim to extend the PIP concept to other filters as well, such as unsharp masking filter [17].

Acknowledgements

Support for this work includes University of Minnesota Computer Science Department Start-up funds, University of Minnesota Digital Technology Center Seed Grants 2002-4, NSF ACI-0238486 (CAREER), NSF DMS-0528492, and the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement number DAAD19-01-2- 0014. Its content does not necessarily reflect the position or the policy of this agency, and no official endorsement should be inferred.

The Skeleton hand model is from the Stereolithography Archive at Clemson University. The volumetric CT scanned engine data is from General Electric. The Neghip is a VolVis distribution of SUNY Stony Brook. The Bucky ball data set has been created by Dr. Oliver Kreylos at the University of California, Davis.

References

1. Saito, T., Takahashi, T.: Comprehensible rendering of 3-D shapes. In: Proceedings of SIGGRAPH 1990. Volume 24. (1990) 197–206
2. Yuan, X., Chen, B.: Illustrating surfaces in volume. In: Proceedings of Joint IEEE/EG Symposium on Visualization (VisSym'04), the Eurographics Association (2004) 9–16

3. Nvidia: Nv_fragment_program. NVIDIA OpenGL Extension Specifications for the CineFX Architecture (NV30) (2003)
4. Levoy, M.: Display of surfaces from volume data. IEEE Computer Graphics and Application **8** (1988) 29–37
5. Ebert, D., Rheingans, P.: Volume illustration: Non-photorealistic rendering of volume models. In: Proceedings of IEEE Visualization '00. (2000) 195–202
6. Kindlmann, G., Durkin, J.W.: Semi-automatic generation of transfer functions for direct volume rendering. In: Proceedings of IEEE 1998 Symposium on Volume Visualization. (1998) 79–86
7. Kniss, J., Kindlmann, G., Hansen, C.: Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In: Proceedings of IEEE Visualization '01. (2001) 255–262
8. Hertzmann, A.: Introduction to 3d non-photorealistic rendering:silhouettes and outlines. ACM SIGGRAPH 99 Course Notes (Non-Photorealistic Rendering) (1999)
9. Tomasi, C., Manduchi, R.: Bilateral filtering for gray and color images. Proceedings of the 1998 IEEE International Conference on Computer Vision (1998) 836–846
10. Ostromoukhov, V., Hersch, R.D.: Artistic screening. In: Proceedings of SIGGRAPH 1995, ACM Press (1995) 219–228
11. Hauser, H., Mroz, L., Bischi, G.I., Gröller, M.E.: Two-level volume rendering. IEEE Transactions on Visualization and Computer Graphics **7** (2001) 242–252
12. Zhou, J., Hinz, M., Tönnies, K.D.: Focal region-guided feature-based volume rendering. In: Proceedings of First International Symposium on 3D Data Processing Visualization and Transmission. (2002) 87–90
13. Grigoryan, G., Rheingans, P.: Probabilistic surfaces: Point based primitives to show surface uncertainty. In: Proceedings of IEEE Visualization '02. (2002) 147–154
14. Johnson, C.R., Sanderson, A.R.: A next step: Visualizing errors and uncertainty. IEEE Computer Graphics and Application **23** (2003) 6–10
15. Mark, W.R., Glanville, R.S., Akeley, K., Kilgard, M.J.: Cg: a system for programming graphics hardware in a C-like language. ACM Transactions on Graphics (TOG) **22** (2003) 896–907
16. Van Gelder, A., Kim, K.: Direct volume rendering with shading via three-dimensional textures. In: Proceedings of the 1996 Symposium on Volume visualization. (1996) 23–30
17. Luft, T., Colditz, C., Deussen, O.: Image enhancement by unsharp masking the depth buffer. ACM Transactions on Graphics **25** (2006) 1206–1213

Tracking of Individuals in Very Long Video Sequences

P. Fihl[1], R. Corlin[1], S. Park[2], T.B. Moeslund[1], and M.M. Trivedi[2]

[1] Laboratory of Computer Vision and Media Technology
Aalborg University, Denmark
pfa@cvmt.dk, tbm@cvmt.dk

[2] Computer Vision and Robotics Research Laboratory
The University of California, San Diego, USA
parks@ucsd.edu, mtrivedi@ucsd.edu

Abstract. In this paper we present an approach for automatically detecting and tracking humans in very long video sequences. The detection is based on background subtraction using a multi-mode Codeword method. We enhance this method both in terms of representation and in terms of automatically updating the background allowing for handling gradual and rapid changes. Tracking is conducted by building appearance-based models and matching these over time. Tests show promising detection and tracking results in a ten hour video sequence.

1 Introduction

Visual analysis of humans has a number of applications ranging from automatic surveillance systems to extracting pose parameters for realistically character animation in movies. Automatic surveillance systems observe humans at a distance and in various environments. Furthermore, these systems should, as opposed to e.g., motion capture systems, work completely autonomous and for long periods of time.

The foundation of many surveillance systems is a good detection and tracking of humans in a video sequence. These issues have received much attention in the last decade or so [1,2,3,4]. The detection problem (aka the figure-ground segmentation problem) is typically done using shape, motion, depth, background detection, or appearance [5,6,7,8,9,10]. When the scene of interest contains individuals that are allowed to occlude each other, the tracking of individuals is inherently difficult and using an appearance-based model for each individual is often the preferred approach.

In this work we consider situations where occlusion can occur and we therefore follow the appearance-based approach. Our aim is continuous detection and tracking over very long periods of time as opposed to other approaches mostly evaluated on short video sequences. Concretely, we first develop and use an advanced background subtraction algorithm in order to handle the figure-ground segmentation problem. The result is a silhouette of each individual in the scene - section 2. Next we use an appearance-based model to represent each individual. A good model is obtained by using some of the results from research on modeling interacting people. We then present a scheme for matching appearance-based models over time - section 3. In section 4 we present tracking results of several hours of continuous video and in section 5 a conclusion is given.

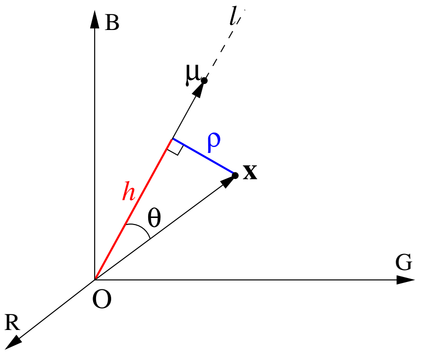
2 Figure-Ground Segmentation

The first step in our tracking algorithm is to separate the foreground (humans) from the background, i.e., the figure-ground segmentation problem. We do this using a background subtracting approach inspired by [11].

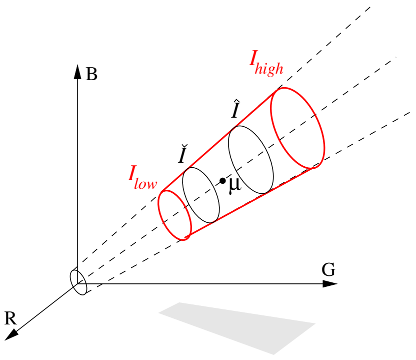
2.1 Background Representation

We apply the Codeword approach of [11], which has shown to perform better than Gaussian mixture models [9] and other well-known methods [12,1] in terms of both speed and sensitivity [13].

The representation of a background pixel in the codeword approach [11] is based on the representation from [14]. Here color and intensity are represented independently and a background pixel is represented as a vector in the RGB-cube, $\boldsymbol{\mu}$. The distance, in terms of color, ρ, from a new pixel, $\boldsymbol{x}$, to the background model is measured as the perpendicular distance to the vector. The difference in intensity is measured along the vector and denoted, h, see figure 1. In the work by [11] a cylinder centered around



(a) The color and intensity representation.



(b) A codeword representation.

Fig. 1. Illustration of the representations used in the background subtraction

the vector represents a codeword for this particle pixel and all pixel values inside the cylinder are classified as background. During a training phase a number of codewords are learned for each pixel and together these are denoted the codebook for this particular pixel. This is a fast and robust approach due to the multi-mode nature of the representation. However, since all color vectors go through the origin of the color-cube a more correct representation is to form a truncated cone around each learned background vector. In this way the different colors inside the codeword actually corresponds to the same colors with different intensity. In figure 1 our representation of a codeword is shown, where $\check{I}$ and $\hat{I}$ are the minimum and maximum values found during training, and I_{low} and I_{high} are where the cone is truncated in order to allow additional values, e.g., due to shadows [15].

2.2 Background Initialization

A key issue in successful background subtraction is to learn a good model of the background during an initialization phase. If no moving objects are present in the scene this is obviously easier. But a more general approach is to allow for moving objects. If a pixel is covered by moving objects in less than 50% of the learning period then a median filter can be applied [1]. A different method is first to divide the training sequence into temporal subintervals with similar values - assumed to belong to the background and then find the subinterval with the minimum average motion and only use these pixels for model initialization [16,17]. In this work we follow the approach by [11], which also works along these lines of reasoning.

During the initialization phase each new pixel is either assigned to an already existing codeword (which is updated accordingly) or a new codeword is created. This will produce codewords for non-background pixels, but these are handled by temporal filtering using the so-called *maximum negative run-length* (MNRL). This measures the longest time interval where no pixel values have been assigned to the codeword. After the training period all codewords with a too large maximum negative run-length will be removed from the background model, i.e., this process allows for moving objects during the training period, see [11] for details.

2.3 Background Updating

Using multiple codewords for each pixel allows modeling of very dynamic scenes, but only the variation that is present in the training period will be modeled by the codebook background method as described in [11][1]. For the background subtraction to work for several hours it is necessary also to handle the changes in the background that occurs after the initialization phase. Two different types of changes need to be handled.

- **Gradual changes** do not change the appearance of the background much from one frame to the next. The accumulated change over time can however be large, e.g., the effect of the changing position of the sun during a day.
- **Rapid changes** cause significant changes in the background from one frame to the next. Background objects that are moved or significant changes in the motion patterns of vegetation caused by gusting winds will for example cause rapid changes.

To handle the gradual changes we apply a simple continuous model:

$$\boldsymbol{\mu}_{t+1} = (1-\alpha)\cdot\boldsymbol{\mu}_t + \alpha\cdot\boldsymbol{x}_t \quad , \quad 0 \le \alpha \le 1 \tag{1}$$

where μ_t is the codeword, μ_{t+1} is the updated codeword, and x_t is the current pixel value. Based on experiences in dynamic outdoor environments α is typical 0.05 to 0.15.

Only the activated codeword in each codebook is updated with this process. Consider the situation where a pixel is sometimes occupied by a green tree branch and sometimes occupied by a red wall, e.g., due to wind. Only the codeword modeling the

[1] It should be noted that at the time when this work was finished [15] the authors of [11] published a more advanced version of their codebook algorithm [18], which is somewhat similar to our background subtracting approach.

branch should be updated when the branch occupies the pixel, since the color of surfaces with different orientation, texture, and color change in different ways with changing illumination. The change of the codeword that models the wall cannot be found from the color change of the branch.

Pixels falsely classified as background will lead to codewords that are updated to model foreground instead of background. Therefore only pixels describing stable background will be updated. Stable background is defined as a pixel that has been classified as background for the last j frames, where j typically is 10-15. As for α this interval is based on experiences in dynamic outdoor environments. The performance of the background updating is not very sensitive to neither j nor α within these intervals.

Equation 1 cannot handle rapid changes and therefore new codewords are learned during run-time. For example, in a situation when a car is parked in the scene it is treated (correctly) as a foreground object. However, while the car is parked we want it to become part of the background so we can detect new foreground objects that move in front of the car. We do this in the following way. Each time a pixel is classified as foreground we create a new codeword, denoted a *training codeword*. If this codeword has a small MNRL within the next n frames we conclude that this codeword does indeed represent a new background and we make it a *temporary codeword*[2]. If the MNRL is big we delete this training codeword. Temporary codewords that become inactive (measured by their MNRL) are deleted, e.g., if the parked car starts to move again. So in each frame a pixel value is matched against the codewords from the "real" background (learned during initialization), the temporary codewords and the training codewords, in that order. If a match is found, the respective codeword is updated using equation 1.

2.4 Bounding Box Representation

After an image is processed by the background subtraction process we remove noise (false positives) using a median filter. Sometimes false negatives result in the silhouette being split into smaller blobs. We therefore investigate the size and proximity of the bounding boxes of each blob and try to merge them into bounding boxes each representing one human [15]. The silhouettes in the merged bounding boxes are compared to a simple body model to distinguish humans from small blobs of noise. The silhouette of a person can roughly be described by an ellipse, and our body model defines limits for the ratio between the major and minor axes of the ellipse, the slope of the major axis, the fidelity between the ellipse and silhouette, and the area of the silhouette. A silhouette complying with all limits of the body model will be accepted as a person otherwise it is considered as noise. To avoid the problem with a person producing multiple enter/exit results when the area of the silhouette is close to the limit of the body model we utilize a hysteresis threshold [15]. To get a correct initialization of our appearance model we need to ensure that a person is completely inside the field-of-view before we accept the new person and we therefore introduce an entry zone around the image border [15]. To

[2] Pixels that belong to a training codeword are classified as foreground whereas pixels belonging to a temporary codeword are classified as background.

summaries, after the above processes we are left with a number of bounding boxes each containing the silhouette of one person.

3 Tracking

3.1 Representation

We model the appearance of a person by dividing it into two regions which are modeled separately: the upper body (not including the head) and the lower body [4]. Due to the nature of the method the regions are not simply found as a ratio of the bounding box as seen in, e.g., [10,19,20] but are found by dividing the body into a set of blobs that are similar in color and spatially connected, and then grouping these blobs into an upper body and a lower body using a ratio of the bounding box as a guideline.

The blobs are initialized by labeling pixels with similar color in the foreground to the same class. To do this the Expectation Maximization (EM) algorithm is used to first learn the gaussian distributions of color classes in the foreground followed by a classification of the pixels to these classes. The labeling of pixels to color classes is carried out by a Maximum Likelihood estimation.

When the pixels have been classified in this way the classes are not necessarily spatially connected, e.g. the dark hair of a person could be assigned to the same class as a dark pair of shoes or simply a checkered shirt could consists of many spatially disconnected classes of the same color. To make sure that each blob represents a region of connected pixels a relabeling is done by making a connected component analysis. This is done by finding the contours of all disconnected regions for the pixels in each color class separately and giving all pixels within the boundary of these contours a unique label.

To avoid over-segmentation of the foreground similar blobs are merged. Blob similarity is evaluated using four criteria [15] and two blobs are merged if either the first criterion is true or if the three remaining criteria are all true: 1) a blob is completely surrounded by another blob, 2) two blobs are adjacent, 3) two blobs share a large border, 4) two blobs are similar in terms of color. By use of these criteria the number of blobs is reduced considerably and a set of blobs that are expected to represent relatively stable parts of the foreground is obtained.

To define the merged blobs as either upper body or lower body we use ratios of the bounding box as a guideline. The bounding box is divided into three regions as shown in figure 2 [15]. All blobs with centroid in the range from 0 to 0.55 times the height of the bounding box will define the lower body and blobs with centroid from 0.55 to 0.84 times the height of the bounding box will define the upper body. This way the border between upper and lower body will not be a straight line but follow the borders of dissimilar blobs. The final features representing a person (or silhouette) are listed in the feature vector $\boldsymbol{m}$:

$$\boldsymbol{m} = [\mu_x, \mu_y, \mu_{H\_upper}, \mu_{S\_upper}, \mu_{H\_lower}, \mu_{S\_lower}]^T \tag{2}$$

where μ_x and μ_y are the mean position or center of mass of the given person. The last four parameters represent the mean color of the upper and lower body respectively in terms of hue and saturation [15].

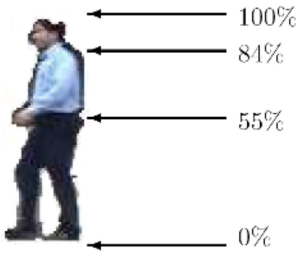


Fig. 2. Ranges for partitioning the silhouette of a person into head, upper body and lower body in relation to the height of the person. Note that the hand is assigned to the upper body even though its located below the 0.55 line. This is due to the complex merging process described above.

3.2 Matching

The matching of identities is performed by calculating the dissimilarity in terms of the Mahalanobis distance[3] between all extracted silhouettes in the current frame, indexed by i, and all known identities that have been tracked to this frame, indexed by j:

$$\Delta_{ij} = (\boldsymbol{m}_i - \boldsymbol{m}_j)^T (\Pi_i + \Pi_j)^{-1} (\boldsymbol{m}_i - \boldsymbol{m}_j) \tag{3}$$

where Π_i represents the between class covariance of all body models in the current frame and Π_j the between class covariance for the identities that the body models are being matched to. These are pooled in order to compensate for the differences in the variations of the body models and the identities they are being matched to. To simplify the calculations only the diagonal of these covariances have been used. The between class variances have been calculated as follows:

$$\Pi = \frac{1}{k} \sum_k (\boldsymbol{\mu}_k - \boldsymbol{\mu}_{all})(\boldsymbol{\mu}_k - \boldsymbol{\mu}_{all})^T \tag{4}$$

where $\boldsymbol{\mu}_k$ is the mean value of the k^{th} body model/identity and $\boldsymbol{\mu}_{all}$ is the mean of all body models/identities present in the given region [15].

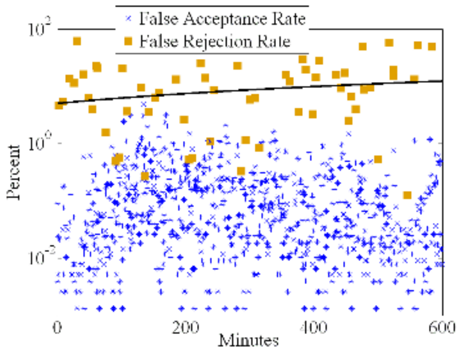
4 Results

The presented system has been tested at two levels. The figure-ground segmentation have been tested to show its performance on very long video sequences (10 hours). The bounding box representation and the tracking have been tested on the output of the figure-ground segmentation to show the system's overall capability to track people.

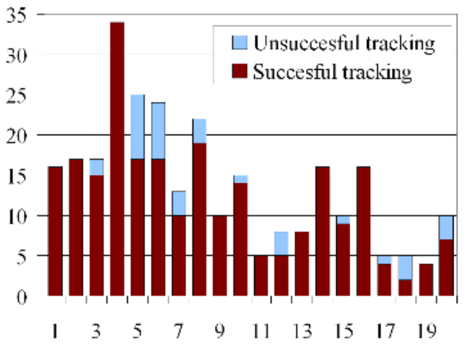
4.1 Test of Figure-Ground Segmentation

The video used for the test is a 10 hour video with a frame rate of 30 frames per second. The video is captured from 9.15 AM to 7.15 PM. and the scene contains several challenging

[3] Note that a weight is assigned to each feature in order to balance the positional features and the appearance features. I.e., 0.3 for the positional features and 0.1 for the appearance features. The weights are omitted from the equation for clarity.



(a)



(b)

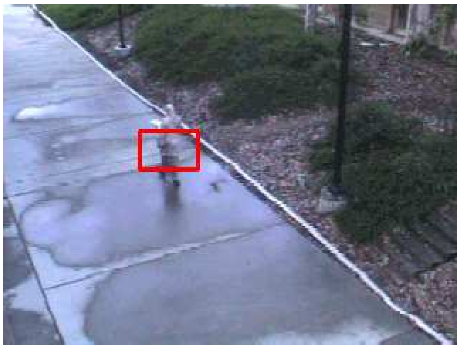
Fig. 3. Results from the 10 hour test video. (a): The FAR and FRR in percent as a function of time. The black line is the best linear fit of the FRR samples in the least-squares sense. Note that the y-axis is logarithmic. (b): The number of successful and unsuccessful tracks. Each column covers 30 minutes.

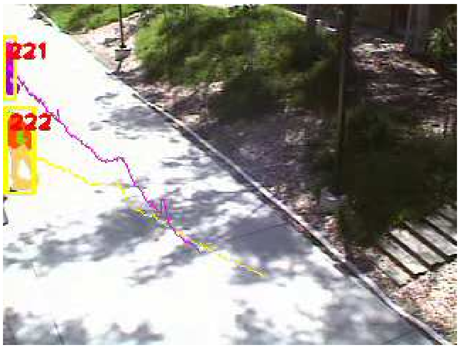
situations in the context of figure-ground segmentation, i.e. illumination changes, non-static background, shadows, puddles, and foreground camouflage (see figure 4).

To calculate the false rejection rate (percentage of foreground pixels falsely classified as background) and the false acceptance rate (percentage of background pixels falsely classified as foreground) a set of 93 frames containing foreground objects (people) have been sampled from the whole time span. The images used are the binary foreground mask obtained from the figure-ground segmentation filtered with the median filter. For each of the sampled frames the foreground region was marked by hand (based on the original input frame) and used as the ground truth.

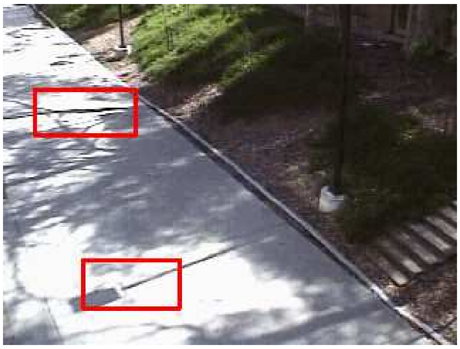
The calculation of the false acceptance rate is based on the above mentioned frames in addition to a set of frames containing only background. The frames containing only background were added since only a limited number of frames actually contain people, and the additional frames would give a more representative result for the whole video. The frames containing only background were sampled every 1000 frames. When sampling every 1000 frames some of the frames contained people, but these frames were discarded from the set giving a total of 971 frames.

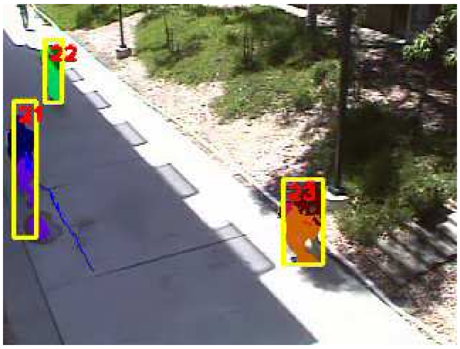
Figure 3(a) shows FRR (false rejection rate) and FAR (false acceptance rate) in percent as a function of time. FRR is in the range [0%;62.4%] with mean value 8.45% and standard deviation 13.43. The black line represents the best linear fit of the FRR samples in the least-squares sense and shows a slightly increasing tendency in FRR. The increase does however not mean that the performance of the background subtraction decreases over time. The increase in FRR is caused by the low overall illumination level at the end of the day which causes the problem of foreground camouflage to increase. The FAR is in the range [0%;4.9%] with mean value 0.14% and standard deviation 0.33. The FAR shows a slightly decreasing tendency over time. The mean FAR of 0.14% shows that the background subtraction in general effectively models the background and adapts to the changes present in the test video. The performance of the background subtraction in terms of FAR in not dependent on how many hours it has been running,


(a) 9.25 AM. The box shows an example of a region with foreground camouflage. Notice the puddles on the ground.

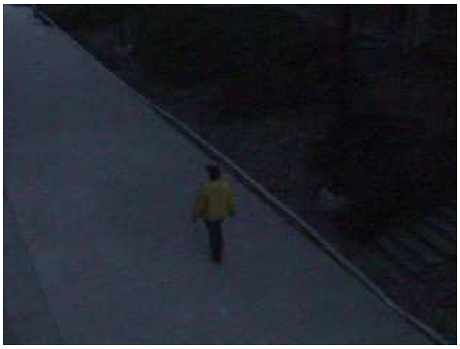


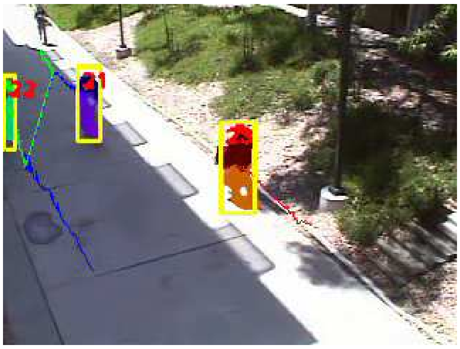
(b) The two persons are tracked correctly even though they move near each other.


(c) 3.05 PM. The boxes show examples of regions with strong shadows. Furthermore, other shadows move rapidly because of the wind.



(d) The three people are tracked correctly until the two persons to the left get too close to each other.


(e) 7.05 PM. The overall illumination level changes significantly from morning to the afternoon and again from the afternoon to the evening



(f) The tracks of the two persons to the left are swapped.

Fig. 4. Left: Examples of the changes in the scene during the 10 hours test video. Right: Tracking results.

but on the type of changes that happens in the scene, and the background subtraction automatically recovers from changes that are not directly handled by the model.

4.2 Test of Tracking

The 10 hours of test video contains 267 persons that move through the scene[4]. The tracking result of each person has been evaluated to see if the system successfully identifies each person and tracks the person.

The system identifies and tracks 247 persons successfully. 20 persons are not tracked correctly and the system further identifies 15 blobs of noise as persons which gives a total of 35 errors. The overall successful tracking rate yields 86.9%. Figure 3(b) shows the tracking result for each 30 minutes interval.

Figure 3(b) indicates that the performance of the tracking is independent of the number of hours the system has been running. The number of tracking errors is most remarkable in the 3rd and 9th hour. This is due to rapid background variations and low overall illumination, respectively, and not because the system has been running for several hours. Figure 4 shows examples of tracking results (both successful and unsuccessful) when multiple people are in the scene.

The errors that occur during tracking can be explained by either *noisy foreground segmentation* (24 errors) or *insufficient tracking or appearance model* (11 errors). The errors originating from noisy foreground segmentation are mainly due to moving vegetation and strong shadows that gets identified as humans. The errors originating from insufficient tracking or appearance model often occur when two persons move close past each other (resulting in switching identities) or when strong shadows makes the silhouette of a person non-elliptic. The effect of both types of errors can possibly be reduced by including temporal information.

5 Conclusion

In this paper we have presented a system to do figure-ground segmentation and tracking of people in an outdoor scene continuously for several hours.

The system has been thoroughly tested on 10 hours of continuous video containing multiple difficult situations. The system was able to automatically update the background model allowing for tracking people with a success rate of 86.9%, and we believe that this number can be increased with relatively simple improvements to the tracking algorithm. To our knowledge this is the first system to present results on continuous tracking in very long video sequences.

References

1. Haritaoglu, I., Harwood, D., Davis, L.: W4: Real-Time Surveillance of People and Their Activities. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000)
2. McKenna, S., Jabri, S., Duric, Z., Wechsler, H.: Tracking Interacting People. In: The fourth International Conference on Automatic Face and Gesture Recognition, Grenoble, France (2000)

[4] People that never enter the field-of-view completely or people that move through the scene in groups are not included in this number or the test.

3. Zhao, T., Nevatia, R.: Tracking Multiple Humans in Crowded Environments. In: Computer Vision and Pattern Recognition, Washington DC, USA (2004)
4. Park, S., Aggarwal, J.: Simultaneous tracking of multiple body parts of interacting persons. Computer Vision and Image Understanding **102** (2006)
5. Leibe, B., Seemann, E., Schiele, B.: Pedestrian Detection in Crowded Scenes. In: Computer Vision and Pattern Recognition, San Diego, CA, USA (2005)
6. Viola, P., Jones, M., Snow, D.: Detecting Pedestrians Using Patterns of Motion and Appearance. International Journal of Computer Vision **63** (2005)
7. Sidenbladh, H.: Detecting Human Motion with Support Vector Machines. In: International Conference on Pattern Recognition, Cambridge, UK (2004)
8. Hayashi, K., Hashimoto, M., Sumi, K., Sasakawa, K.: Multiple-Person Tracker with a Fixed Slanting Stereo Camera. In: International Conference on Automatic Face and Gesture Recognition, Seoul, Korea (2004)
9. Stauffer, C., Grimson, W.: Adaptive Background Mixture Models for Real-Time Tracking. In: Computer Vision and Pattern Recognition, Santa Barbara, CA, USA (1998)
10. Roth, D., Doubek, P., Gool, L.: Bayesian Pixel Classification for Human Tracking. In: IEEE Workshop on Motion and Video Computing (MOTION'05), Breckenridge, Colorado (2005)
11. Kim, K., Chalidabhongse, T.H., Harwood, D., Davis, L.: Background modeling and subtraction by codebook construction. In: IEEE International Conference on Image Processing (ICIP). (2004)
12. Elgammal, A., Harwood, D., Davis, L.: Non-Parametric Model for Background Subtraction. In: European Conference on Computer Vision, Dublin, Ireland (2000)
13. Chalidabhongse, T., Kim, K., Harwood, D., Davis, L.: A Perturbation Method for Evaluating Background Subtraction Algorithms. In: Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, Beijing, China (2005)
14. Horprasert, T., Harwood, D., Davis, L.: A Statistical Approach for Real-time Robust Background Subtraction and Shadow Detection. In: IEEE ICCV'99 FRAME-RATE WORKSHOP, Corfu, Greece (1999)
15. Andersen, P., Corlin, R.: Tracking of Interacting People and Their Body Parts for Outdoor Surveillance. Master's thesis, Laboratory of Computer Vision and Media Technology, Aalborg University, Denmark (2005)
16. Gutchess, D., Trajkovic, M., Solal, E., Lyons, D., Jain, A.: A Background Model Initialization Algorithm for Video Surveillance. In: International Conference on Computer Vision, Vancouver, Canada (2001)
17. Wang, H., Suter, D.: Background Initialization with a New Robust Statistical Approach. In: Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, Beijing, China (2005)
18. Kim, K., Chalidabhongse, T., Harwood, D., Davis, L.: Real-Time Foreground-Background Segmentation using Codebook Model. Real-Time Imaging **11** (2005)
19. Yang, C., Duraiswami, R., Davis, L.: Fast Multiple Object Tracking via a Hierarchical Particle Filter. In: International Conference on Computer Vision, Beijing, China (2005)
20. Mittal, A., Davis, L.: M2Tracker: A Multi-View Approach to Segmenting and Tracking People in a Cluttered Scene. International Journal of Computer Vision **51** (2003) 189–203

A Natural Interface for Sign Language Mathematics

Nicoletta Adamo-Villani, Bedřich Beneš, Matt Brisbin, and Bryce Hyland

Purdue University, West Lafayette 47907, USA

Abstract. The general goal of our research is the creation of a natural and intuitive interface for input and recognition of American Sign Language (ASL) math signs. The specific objective of this work is the development of two new interfaces for the Mathsigner^{tm} application. Mathsigner^{tm} is an interactive, 3D animation-based game designed to increase the mathematical skills of deaf children. The program makes use of standard input devices such as mouse and keyboard. In this paper we show a significant extension of the application by proposing two new user interfaces: (1) a glove-based interface, and (2) an interface based on the use of a specialized keyboard. So far, the interfaces allow for real-time input and recognition of the ASL numbers zero to twenty.

1 Introduction

Deaf education, and specifically math/science education, is a pressing national problem [1,2]. To address the need to increase the abilities of young deaf children in math, we have recently created an interactive computer animation program (Mathsigner^{tm}) for classroom and home learning of K-3 (Kindergarten to third grade) arithmetic concepts and signs [3]. The program, currently in use at the Indiana School for the Deaf (ISD), is a web/CD-ROM deliverable desktop application aimed at increasing the opportunity of deaf children to learn arithmetic via interactive media, and the effectiveness of hearing parents in teaching arithmetic to their deaf children. The application includes 3D animated signers that teach ASL mathematics through a series of interactive activities based on standard elementary school math curriculum. The user interacts with the application and responds to questions using mouse and keyboard.

Based on feedback collected from ISD teachers, parents and students, and from signers who have tested the application extensively, the current interface presents various limitations.

1. Young deaf children of deaf parents are likely to know the signs for the numbers but might not be familiar yet with the corresponding math symbols. In this case, the children should be able to enter the answer to a problem by forming the correct ASL hand shapes, rather than by pressing a number key.
2. Deaf children of hearing parents use the application not only to increase their math skills, but also to learn the correct signs for math terminology.

Presently, the program does not allow the students to test and get feedback on their signing skills since all interactive activities require responses in the form of mouse clicks and/or keystrokes.

3. Hearing parents, undertaking the study of the ASL signs for math terminology, can only test their ability to recognize the signs; they do not have the opportunity to self test their ability to produce the signs correctly (it is common for beginner signers to perform the signs with slight inaccuracies).

In an effort to improve on the current implementation of the program, we propose two new user interfaces which allow for real-time hand gesture input and recognition. Interface (1) uses an 18-sensors Immersion cyberglove [4] as the input device. The user wears the glove and inputs an ASL number in response to a particular math question (for instance, '8' in response to question'3+5=?'). A pre-trained neural network detects and recognizes the number sign. The result is sent to the Mathsigner^{tm} application which evaluates the answer to the question and gives feedback to the user.

Interface (2) (currently under development) is based on the use of a recently developed human-computer communication method for keyboard encoding of hand gestures (KUI) [5], and a specialized keyboard for gesture control [6]. The KUI method allows for input of any hand gesture by mapping each letter key of the keyboard to one degree of freedom of a 3 dimensional hand. Each hand configuration is visualized in real-time by the use of a 3D hand model, and encoded as an alphanumeric string. Hand posture recognition and communication with the Mathsigner^{tm} are implemented as in interface (1).

In Section 2 of the paper we present a brief overview of current approaches in sign language input and recognition. In Section 3 we describe the two new user interfaces in detail, and in Section 4 we discuss their merits and limitations, along with future work. Conclusive remarks are presented in the last section.

2 Background

'Computer technology offers the opportunity to create tools that enable literacy and learning in ways accessible to signing users' [7]. In order to be effective, these tools need to support sign language interfaces, i.e., ways of input, recognition, and display of signing gestures.

Sign language input and recognition has been an active area of research during the past decade. Currently, there are two main approaches to gesture input: direct-device and vision-based input [8,9,10]. The direct-device approach uses a number of commercially available instrumented gloves, flexion sensors, body trackers, etc. as input to gesture recognition [11,12]. Some advantages of direct devices, such as data gloves, include: direct measurement of hand and finger parameters (i.e., joint angles, wrist rotation and 3D spatial information), data input at a high sample frequency, and no line-of-sign occlusion problems. Disadvantages include: reduced user's range of motion and comfort and high cost of accurate systems (i.e., gloves with a large number of sensors –18 or 22–).

Vision based approaches use one or more video cameras to capture images of the hands and interpret them to produce visual features that can be used to recognize gestures. The main advantage of vision-based systems is that they allow the users to remain unencumbered. Main disadvantages include: complex computation requirements in order to extract usable information, line-of sign occlusion problems, and sensitivity to lighting conditions.

Recently, researchers have started to develop gesture input systems that combine image- and device- based techniques in order to gather more information about gestures, and thereby enable more accurate recognition. Such hybrid systems are often used to capture hand gestures and facial expressions simultaneously [13].

Recognition methods vary depending on whether the signs are represented by static hand poses or by moving gestures. Recognition of static signing gestures can be accomplished using techniques such as template matching, geometric feature classification, neural networks, or other standard pattern recognition methods to classify the pose [14]. Recognition of dynamic gestures is more complex because it requires consideration of temporal events. It is usually accomplished through the use of techniques such as time-compressing templates, dynamic time warping, Hidden Markov Models (HMMs) [15,16] and Bayesan Networks [17].

In this paper we are concerned with static or semi-static ASL gestures. The goal is input and recognition of ASL numbers 0-20 which are represented by static hand-shapes (numbers 0-9) and by hand gestures requiring a very limited range of motion (numbers 10-20) [2,18]. To capture the hand gestures, we have chosen a direct-device approach because research findings show that this approach yields more accurate results [19]. The specialized keyboard of interface (2) is not a whole-hand input device since the input is not derived from direct measurements of hand motions, but from measurements of the motions (keystrokes) of a device manipulated by the hand. However, the keyboard allows for intuitive and natural input of hand gestures if we consider that the layout of the key sites corresponds to the layout of the movable joints of the hand (see Figure 4). Thus, we can think of the specialized keyboard as a 'semi direct' input device.

3 Implementation

3.1 Interface (1): Glove-Based

This interface makes use of a light-weight Immersion cyberglove which provides 18 angles as inputs. The glove has two bend sensors on each finger, four abduction sensors, and sensors for measuring thumb cross-over, palm arch, wrist flexion, and wrist abduction. To recognize the sign gesture input via the glove, we have used two approaches: (1) a basic metric measure in the space of the possible glove configurations, and (2) neural networks.

Distance Metrics. For this approach, five signers used the glove to input the ASL numbers 0-20 once. A stand alone program developed in C++ was used to capture and store the hand-shapes for later comparison. During interaction within the Mathsigner^{tm}, the C++ application compares the distance measures of the input gesture to the pre-stored ones. The distance measure is the classical Euclidian metrics, where each of the two angles α and α' is compared as:

$$dist = \sqrt{(\alpha - \alpha')^2}.$$

This test is performed for each angle. If the distance measures fall within the sensitivity level, the hand shape is recognized. Based on the first-fail test, if any distance measure is larger than the sensitivity level, the hand-shape is not matched to any of the gestures in the training data set. The experimentally set level was 30^o. With this method, while speed of response was fairly high (20kHz), recognition accuracy with unregistered users (i.e., users not represented in the training data set) was low. This is due primarily to variations in users' hand size. The neural networks approach, described in the next section, provided a better solution.

Neural Networks. This approach is based on the Fast Artificial Neural Network Library, (FANN) [20] a freely available package from Sourceforge. This library supports various configurations of neural networks. We have experimented with the following two configurations. The first one is based on a single neural network for all signs, whereas the second one uses different neural networks for different signs. The first configuration involves 18 neurons on the input and 21 on the output. The input neurons correspond to the input angles from the data glove. The 21 output values define 1-of-21 possible hand gestures. While this configuration yielded fairly accurate recognition results, it did not provide high speed of recognition. The configuration described in the next paragraph provides higher accuracy rate and real-time recognition.

This configuration is the standard complete backward propagation neural network with symmetrical sigmoid activation function [20]. Instead of using one neural network, it uses a set of networks (one per sign) with 18 input neurons that corresponds to the 18 angles from the data glove. One output neuron for each network determines whether the input configuration is correct (value close to 1) or incorrect (value close to -1 because of symmetrical sigmoid function). Each neural network uses two hidden layers of completely connected neurons, each layer containing 25 neurons (see Fig. 1). The training error was set to 10^{-6} and training of all 21 neural networks for all the input sets was realized in about 10 minutes on a standard laptop with 1.6 GHz Intel Pentium. The neural networks were correctly trained after not more than 10^4 epochs.

The detection of one sign was, on the same computer, performed at the rate of about 20Hz . The accuracy rate with registered users was 90%.The accuracy rate with three unregistered users was 70%. The relatively poor performance for unregistered users is probably due to the small training set of the neural network.

Sign detection is described by the following pseudocode. It is important to note that the signs 0-10 are represented as a single sign, while numbers greater than 10 are represented as a sequence of two signs.

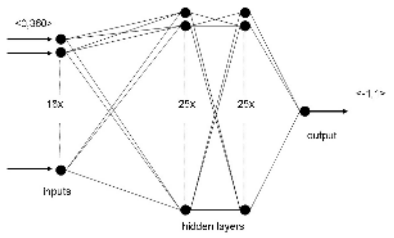


Fig. 1. The neural network has 18 inputs in the input layer, two hidden layers with 25 neurons, and 1 output neuron. This network recognizes one sign.

```
1. Load all trained neural networks $a[i]$.
2. Until the end of the simulation
   (a) Read the data from the data glove
   (b) for (i=0;i<10;i++) process the data with the $a[i]$.
       Remember the index of the maximum.
   (c) If the maximum is greater than 10, read the following sign and process
       it in the same way. The two signs define the number.
   (d) Send the recognized number to the Mathsigner$^{tm}$.
3. Destroy networks, free memory
```

Training. The training data set was provided by five signers. Each signer input the hand shapes corresponding to ASL numbers 0-20 three times. The training data set for each number is composed of 3×5 correct signs and 15 incorrect signs. The training set for each number includes the 15 ASL handshapes corresponding to that number, and 15 randomly selected ASL configurations corresponding to different numbers (provided by the same signers).

Communication with Mathsigner^{tm}. Continuous communication between the cyberglove (or the specialized keyboard, for interface (2)) and the Mathsigner^{tm} application (developed in Macromedia Director MX) was implemented using a built-in Lingo function that allows access to the system 'clipboard'. After sign recognition occurs, the C++ application formats and copies the number corresponding to the ASL hand gesture to the 'clipboard'.

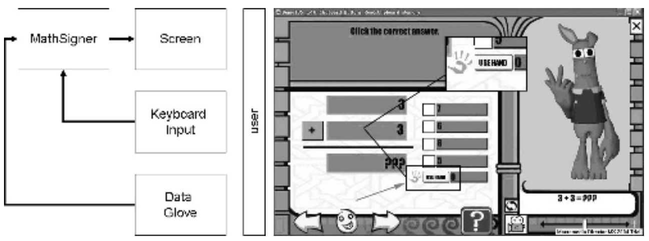
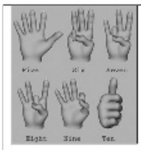


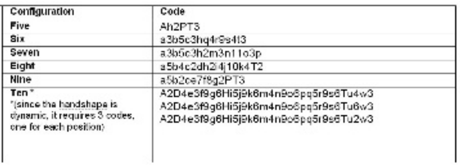
Fig. 2. Schema of the system, left; screenshot of the Mathsigner^{tm} with arrow pointing to the button (and hand icon) used to input the sign answer, right

Within Mathsigner^{tm}, the value retrieved from the 'clipboard' is displayed to the student. To submit an answer to a mathematical question, the student has two options. The student can mouse-click on one of four possible answers, only one being correct; or the student can press the 'Use Hand' button and submit the signing gesture corresponding to the answer (see Fig. 2). Upon submission of the answer, the 3D avatar signs whether the student's response is right or wrong. A video illustrating the use of interface (1) is available at http://www2.tech.purdue.edu/cgt/I3/mathinterface/.

3.2 Interface (2): Keyboard-Based

This interface makes use of a recently developed keyboard-based method for input, modelling, and animation of hand gestures (KUI) [5]. KUI is based on the realization that a hand gesture path requires the same number (26) of parameters as the letters of the English alphabet, thus, via keyboard input, it is possible to enter any hand pose in real-time. By touching a letter key, the user rotates the corresponding joint of a 3D hand a pre-specified number of degrees around a particular axis. The rotation 'quantum' induced by each keystroke can be easily changed to increase or decrease hand configuration precision. The keystrokes corresponding to particular hand poses are recorded and reduced to alphanumeric compact codes; the codes can be used for hand gesture recognition, or as keyframes to produce animation sequences. Figure 3 shows the ASL handshapes for numbers 8-10 produced with the KUI method, and their corresponding alphanumeric codes (the rotation 'quantum' was set to 10 degrees for finger flexion and 5 degrees for finger abduction).





Configuration	Code
Five	Ah2PT3
Six	a3b5c3hq4r9s4l3
Seven	a3b5c3h2m3n1 1o3p
Eight	a5b4c2dh2i4j10k4T2
Nine	a5b2ce7f8g2PT3
Ten * *(since the handshape is dynamic, it requires 3 codes, one for each position)	A2D4e3f9g6Hi5j9k6m4n9o6pq5r9s6Tu4w3 A2D4e3f9g6Hi5j9k6m4n9o6pq5r9s6Tu6w3 A2D4e3f9g6Hi5j9k6m4n9o6pq5r9s6Tu2w3

Fig. 3. Alphanumeric codes for ASL numbers 8-10

Recently, the KUI method was developed into a more powerful technique by the realization of a specialized, reconfigurable keyboard whose layout approximates the projection of the hand joints locations on a plain [6]. The keyboard is shown in Fig. 4.

With this keyboard, the signer inputs a hand gesture by mimicking the fingers' motion of a hand guiding another hand placed under it. The hand configuration, represented by the alphanumeric code, is visualized in real-time in a floating window. When the user is satisfied with the hand-shape, she/he clicks on the 'hand button' (see Fig. 2) in the Mathsigner^{tm} application. The alphanumeric code is converted to joint angles and recognition and communication with the Mathsigner^{tm} are achieved as in interface (1). For this interface, the training

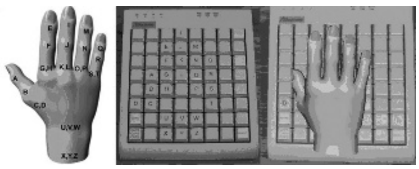


Fig. 4. Alphabetical code for the hand joints, left; specialized keyboard, center; position of hand over keyboard, right

data set was provided by five signers who used the specialized keyboard to input ASL number configurations 0-20 three times.

4 Discussion

Both interfaces have their own merits and limitations. The main advantage of interface (1) lies in allowing the user to input signs in a natural way, without intermediary devices. Another merit is high speed of sign recognition and accuracy rate. One drawback is the high cost of the cyberglove due to the large number of sensors required to input the hand gesture with precision. Currently, the cost of the glove is a major obstacle to immediate dissemination of the program to parents and children for testing at home, and for future commercialization of the application. We are investigating more affordable types of gloves available on the market (http://www.vrealities.com/glove.html) or created by researchers specifically for input of signing gestures [21,22].

The main advantage of interface (2) is the low cost of the keypad. In addition, even if interface (2) is still under development, we anticipate higher accuracy level since variation in hand size is not an issue. The main drawback is that the specialized keyboard is not a true direct input device like the glove. While input of finger flexion (pitch rotations) is fairly natural and intuitive, input of finger abduction (yaw rotations) and wrist rotation and translation (position and orientation of the hand in 3D space) requires a certain degree of learning, abstraction, and practice. The research team is currently working on development of a new hand shaped keyboard which has an 'anatomical cradle' to support the hand, and which allows for more intuitive input of fingers' yaw rotations.

Presently, a limitation of both interfaces is that recognition is restricted to ASL numbers 0-20. In order to enable the user to answer any math question included in the application, input and recognition need to be extended to include numbers 1-1000, decimals, fractions, and the finger-spelling alphabet. In addition, one characteristic of ASL numbers is that they are signed in different ways depending on their meaning (i.e., numbers used to describe quantities–cardinals–, numbers for monetary values, numbers associated with tell-time activities, etc.) [23]. For instance, for dollar numbers 1-9, the number hand-shape is associated with a twisting motion (wrist roll) to indicate dollars. In future implementations of the interfaces, the recognition system will consider these variations.

Many aspects of the interfaces still need to be tested and improved. A comparative evaluation of the interfaces will be carried out in Fall 2006 at ISD with deaf children, parents, and ASL teachers. Besides assessing the usability of the interfaces, the full-scale evaluation will address the problem of signer-independent recognition. An ideal sign recognition system should give good recognition accuracy for signers not represented in the training data set [24]. Inter-person variations that could impact sign recognition include different signing styles, different sign usage due to geographical and social background, and fit of gloves. Many works report that recognition accuracy for unregistered signers decreases severely (by 30- 40%) when the number of signers in the training set is small, and when the signs involve significant, continuous movement [24]. For interface (1), we are concerned with the problem of degradation of recognition accuracy due to fit of the gloves, but we anticipate good recognition results considered that many of the signs are static or involve minimal motion. Studies show that recognition accuracy for unregistered signers is relatively good when only hand shapes and/or limited motion are considered [25]. So far, three unregistered signers have used interface (1); recognition accuracy was 70%.

5 Conclusions

The interfaces presented in this paper are still to be considered prototypes since many of their features are only at a first stage of development. But in spite of their limitations, they are, to our knowledge, the first sign language interfaces specifically designed for input and recognition of ASL signs for mathematics. One interface includes an 18-sensor cyberglove as the input device, and makes use of neural networks for sign recognition. The other interface uses a specialized keyboard for input of signing gestures, and neural networks for recognition.

Many applications to math and science education of the Deaf are conceivable using these interfaces, even at this stage of development. Applications to Virtual Environments are easy to envision. For example, future work involves adapting the glove-based interface for navigation and gesture input/ recognition within an Immersive Virtual Learning Environment that we have recently developed for deaf children [26].

In conclusion, research findings show that automatic analysis of Sign Language gestures has come a long way, and current work can successfully deal with dynamic signs which involve movement and which appear in continuous sequences [24]. However, much remains to be done before sign language interfaces may become commonplace in face to face computer human interaction. Aspects of gesture recognition that need further investigation and attention are building signer-independent recognition systems, and addressing the most difficult aspects of signing, such as grammatical inflections and mimetic signs, and non-manual signals (NMS). While interpretation of NMS in conjunction with gesture recognition is fundamental for understanding sign language communication in general [27], it is not so important for ASL mathematics. Therefore, considered that most ASL mathematics signs are represented by static or semi-static signs, and do not rely greatly on NMS, we believe that the realization of a natural American Sign Language interface for mathematics is a goal achievable in the near future.

Acknowledgments

This research is partially supported by the College of Technology at Purdue University (I3 grant - Proposal #00006585 - http://www.tech.purdue.edu/cgt/I3/), by the Envision Center for Data Perceptualization, and by PHS-NIH Nat. Inst. Deafness, Comm. Disorders award # 5 R01 DC005241-02. We thank Professor Ronnie Wilbur from the Department of Audiology and Speech Sciences at Purdue University for her valuable contributions to the project. In addition, we are grateful to all the signers who have participated in the development and initial evaluation of the interfaces, and to the Indiana School for the Deaf (ISD) for providing the testing ground for a thorough assessment of the interfaces in Fall 2006.

References

1. Holt, J.A., Traxler, C., Allen, T.: Interpreting the scores: A user's guide to the 9th edition stanford achievement test for educators of deaf and hard-of-hearing students. Technical report, Gallaudet Research Institute, Washington, D.C (1997)
2. Caccamise, F., Lang, H.: Signs for Science and Mathematics: A Resource Book for Teachers and Students. Rochester, NY, National Technical Institute for the Deaf, RIT (1996)
3. Adamo-Villani, N., Doublestein, J., Martin, Z.: The mathsigner: An interactive learning tool for american sign language k-3 mathematics. In: IEEE Proceedings of IV04 - 8th International Conference on Information Visualization, Los Alamitos, CA, USA, IEEE Computer Society (2004) 713–716
4. Corporation, I.: 3d interaction. http://www.immersion.com/3d/products/cyber_glove.php (2001)
5. Adamo-Villani, N., Beni, G.: Keyboard encoding of hand gestures. In: Proceedings of HCI International - 10th International Conference on Human-Computer Interaction. Volume 2. (2003) 571–575
6. Adamo-Villani, N., Beni, G.: Reconfigurable keyboard for gesture control. In: HCI International-11th International Conference on Human-Computer Interaction, on a CD-ROM (2005)
7. Frishberg, N., Corazzo, S., Day, L., Wilcox, S., Schulmeister, R.: Sign language interfaces. In: Proc. of INTERCHI-93, Amsterdam, The Netherlands (1993) 194–197
8. Huang, T., Pavlovic, V.: Hand gesture modeling, analysis, and synthesis. In: Proceedings of the International Workshop on Automatic Face and Gesture Recognition, Zurich (1995)
9. Geer, D.: Will gesture-recognition technology point the way? Computer **37** (2004) 20–23
10. Yi, B., Jr., F.C.H., Wang, L., Yan, Y.: Real-time natural hand gestures. Computing in Science and Engineering **7** (2005) 92–96, c3

11. Sturman, D.J.: Whole-Hand Input. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA (1992)
12. Sturman, D.J., Zeltzer, D.: A survey of glove-based input. IEEE Comput. Graph. Appl. **14** (1994) 30–39
13. Culver, V.R.: A hybrid sign language recognition system. In: IEEE Proceedings of the 8th International Symposium on Wearable Computers (ISWC04). Volume 00., Los Alamitos, CA, USA, IEEE Computer Society (2004) 30–33
14. Vamplew, P.: Recognition of sign language gestures using neural networks. In: Proc. of 1st Euro. Conf. Disability, Virtual Reality Assoc. Tech., Maidenhead, UK (1996) 27–33
15. Starner, T., Pentland, A.: Real-time american sign language recognition from video using hidden markov models. Technical Report MIT TR-375, Media Lab, MIT (1996)
16. Rabiner, L.: A tutorial on hidden markov models and selected applications in speech recognition. In: Proceedings of the IEEE. Volume 77(1)., Los Alamitos, CA, USA, IEEE Computer Society (1989) 257–286
17. Avils-Arriaga, H., Sucar, L.E.: Dynamic bayesian networks for visual recognition of dynamic gestures. Journal of Intelligent and Fuzzy Systems **12** (2002) 243–250
18. Flodin, M.: Signing illustrated: the complete learning guide. The Berkley Publishing Group, New York (1994)
19. Hernandez-Rebollar, J.L., Lindeman, R.W., Kyriakopoulos, N.: A multi-class pattern recognition system for practical finger spelling translation. Proc. of IEEE 4th Int'l Conference on Multimodal Interfaces (ICMI'02) (2002) 185–190
20. Nissen, S.: Fast artificial neural network library. http://leenissen.dk/fann/ (2000)
21. Hernandez-Rebollar, J.L., Lindeman, R.W., Kyriakopoulos, N.: The acceleglove: a whole hand input device for virtual reality. In: Proc. of ACM Siggraph 2002 - 29th Int'l Conference on Computer Graphics and Interactive Techniques - Sketches and Applications. (2002) 259
22. Kuroda, T., Tabata, Y., Goto, A., Ikuta, H., Murakami, M.: Consumer price data-glove for sign language recognition. In: Proc. of 5th Intl Conf. Disability, Virtual Reality Assoc. Tech., Oxford, UK (2004) 253–258
23. Dawnsign: Numbering in American Sign Language: Number Signs for Everyone. DawnSign Press (1998)
24. Ong, S., Ranganath, S.: Automatic sign language analysis: A survey and the future beyond lexical meaning. IEEE Transactions on Pattern Analysis and Machine Intelligence **27** (2005) 873–891
25. Su, M.: A fuzzy rule-based approach to spatio-temporal hand gesture recognition. IEEE Trans. Systems, Man, and Cybernetics, Part C: Application Rev. **30** (2000) 276–281
26. Adamo-Villani, N., Carpenter, E., Arns, L.: An immersive virtual environment for learning sign language mathematics. In: ACM Proceedings of Siggraph 2006 - 33rd International Conference on Computer Graphics and Interactive Techniques - Educators, ACM Siggraph (2006)
27. Valli, C., Lucas, C.: Linguistics of American Sign Language: a Resource Text for ASL Users. Washington, D.C.: Gallaudet University Press (2002)

A Novel Gait Recognition Method Via Fusing Shape and Kinematics Features

Yanmei Chai, Qing Wang, Jingping Jia, and Rongchun Zhao

School of Computer Science and Engineering,
Northwestern Polytechnical University
Xi'an 710072, P.R. China

Abstract. Existing methods of gait recognition are mostly based on either holistic shape information or kinematics features. Both of them are very important cues in human gait recognition. In this paper we propose a novel method via fusing shape and motion features. Firstly, the binary silhouette of a walking person is detected from each frame of the monocular image sequences. Then the static shape is represented using the ratio of the body's height to width and the pixel number of silhouette. Meanwhile, a 2D stick figure model and trajectory-based kinematics features are extracted from the image sequences for describing and analyzing the gait motion. Next, we discuss two fusion strategies relevant to the above mentioned feature sets: feature level fusion and decision level fusion. Finally, a similarity measurement based on the gait cycles and two different classifiers (Nearest Neighbor and KNN) are carried out to recognize different subjects. Experimental results on UCSD and CMU databases demonstrate the feasibility of the proposed algorithm and show that fusion can be an effective strategy to improve the recognition performance.

1 Introduction

Gait is a new biometric aimed to recognize person via the style of people walking, which contains physiological or behavioral characteristics of human being. A unique advantage of gait is the ability to operate at a distance, when other biometrics is of too low a resolution to be perceived. Moreover, Gait measurements are also non-intrusive and difficult to disguise or conceal in application scenarios (e.g. face can be obscured by helmets and fingerprints can be hidden by gloves) [1]. All these related subjects lend strong support to the potential for gait as a useful biometrics measurement.

Existing methods of gait recognition can be grouped into holistic ones and model-based ones. Holistic approaches aim to extract statistical features from a subject's silhouette to distinguish different walkers. These include averaged silhouette [2], baseline algorithm [3], method based on silhouette representation and PCA [4], continuous HMMs [5] and discrete symmetry operator [1] etc. Model-based approaches, on the other hand, aim to model the motion of the body and the kinematics of joint angles. For example, Cunado et al [6] propose a technique, which considers the legs as an interlinked pendulum and uses the phase-weighted Fourier magnitude spectra as the feature to recognize different subjects. Ning et al [7] extract the dynamic gait signal by tracking the variation of the walker's major joint angles.

These methods suffer, in our opinion, from a common shortcoming: only part of information of gait motion is used, either shape feature or kinematics of joint angles. However, human vision perception system is not only dependent on a single gait feature to recognize a person, for there are many properties of gait that might serve as recognition features. So fusing multiple features can be a strategy to improve the recognition performance.

Usually, the properties of gait can be categorized as shape features and kinematics ones. The former reflect the holistic features of the body such as height and figure, while the latter represent gait motion features such as joint-angle trajectories of main limbs. In this paper, we use the ratio of the body's height to width and the pixel number of silhouette to represent the shape. Meanwhile, a 2D stick figure is used to represent the human body model, and joint-angle trajectories are calculated to describe the gait motion. Then we fuse the above features on the feature level and the decision level respectively. Finally, the NN and KNN classifiers, together with the similarity measurement based on the gait cycles, are carried out to recognize different subjects.

2 Preprocessing

Firstly, there are two assumptions for the human walking sequences in our experiments: (1) The camera is static and the body in the field of view is not occluded by other objects or background, and (2) The image sequence of side-view is used since the gait of a person is best brought out in the side-view.

2.1 Silhouette Extraction

The silhouette extraction is necessary. It is essential to extract the required human body by eliminating irrelevant background from each image. The detailed steps are described as follows.

1. To obtain an approximate background image of a walking sequence, a mean image is computed by averaging gray-level values for each pixel position over the entire image sequence (in Fig.1 (b)).
2. Background subtraction is used to detect moving objects in each frame.
3. Erosion, dilation and component labeling are used to remove small amount of noise, which have been introduced to the binary image map and fill the small hole in the silhouette (in Fig.1(c)).

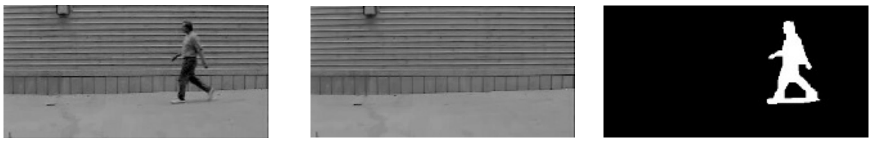
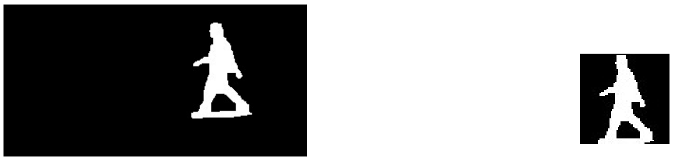

(a) Original image (b) Background (c) Silhouette

Fig. 1. The silhouette extraction

2.2 Image Template

In order to eliminate redundancies and speed up the processing, we need to normalize the segmented images into a scaled template. For each binary silhouette in the sequence, we firstly calculate the centroid, named as (x_c, y_c), and the height and width of segmented object. Then an appropriate length of side L is chosen. Finally, by centering on the centroid (x_c, y_c), we can fit the human silhouette into a fixed $L \times L$ image template. Not only does the normalization carry out across different people, but also it can adjust the changes in scale due to the variation of the distance between the people and camera. The similar work can also be found in [8].



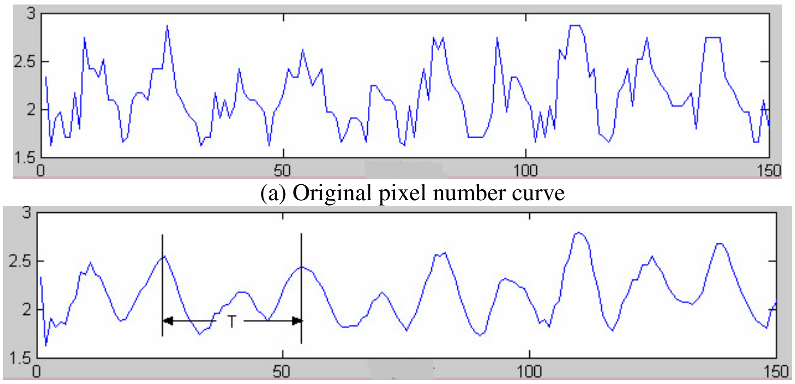
(a) Original image (b) Scaled image template 64×64

Fig. 2. Scaled image template from segmented object

2.3 Gait Periodicity

From the theoretical point of view, human gait is a form of periodic motion, especially when walking laterally. As a result, we can count the number of foreground pixels in the bottom half of the silhouette in each frame over time to estimate the gait periodicity, which is proposed in the paper [2]. This number will reach a maximum when the two legs are farthest apart (full stride stance) and drop to a minimum when the legs overlap (heels together stance). Figure 3 shows an instance of a sequence's pixel numbers curve and the smoothed one by Gaussian filtering.

Notice that two consecutive strides constitute a gait cycle. We compute the median of the distances among minima, skipping every other minimum.



(a) Original pixel number curve

(b) Smoothed curve

Fig. 3. The curves of the pixel number of a gait sequence

3 Gait Signature Extraction

3.1 Kinematics Features Acquisition

To obtain the kinematics features for describing gait motion, we use the body segments properties guided by anatomical knowledge to extract the body main joints position, such as the position of head, neck, shoulder, pelvis, kneel and ankle [9, 10]. It is noticeable that the upper limbs are ignored in our experiment because of the occlusion in the side-view sequence. The body segment properties are shown in Fig.4.

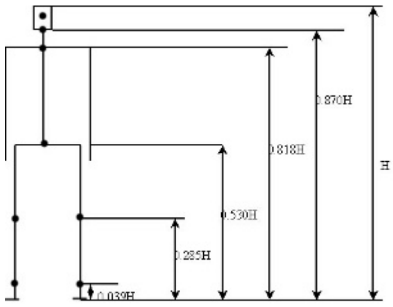


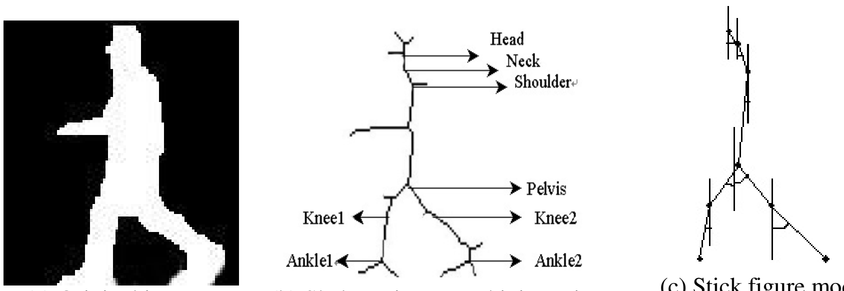
Fig. 4. Body Segment Properties [10]

First of all, we extract the skeleton of binary silhouette image and scan the skeleton image row by row from the top to the bottom. Then the junction of the scan line and skeleton is the joint position. There are 8 coordinates (joint points) in a human body, which are (x_{head}, y_{head}), (x_{neck}, y_{neck}), $(x_{shoulder}, y_{shoulder})$, (x_{pelvis}, y_{pelvis}), (x_{knee1}, y_{knee1}), (x_{knee2}, y_{knee2}), (x_{ankle1}, y_{ankle1}) and (x_{ankel2}, y_{ankle2}) [10]. The skeleton image and joint points are shown in Fig.5 (b). All of these coordinates are connected to form a 2D stick figure model. Furthermore, we can calculate the angle between main limbs and vertical line as the following formula (1). Moreover, there are 7 angles associated with these positions, including θ_{head}, θ_{neck}, θ_{back}, θ_{thigh1}, θ_{thigh2}, θ_{shin1} and θ_{shin2}. The sketch map of stick figure model and the definition of limb angles are show in Fig.5 (c).

$$\theta = \arctan \frac{x_1 - x_0}{y_1 - y_0} \tag{1}$$

where, (x_0, y_0) and (x_1, y_1) are the coordinates of two conjointly joints.

After the beforehand discussion, we find there are 23 dimension dynamic parameters in human stick figure model, which are $2 \times 8 = 16$ dimension ones for joint coordinates and 7 dimension ones for limb angles. Since the x values of joint coordinates are usually fixed, we can ignore them and compress the parameters to 15 dimensions. Furthermore, to eliminate the influence of spatial scale, we normalize these dynamic parameters to a uniform magnitude $[\pi/2, 3\pi/2]$.



(a) Original image (b) Skeleton image and joint points (c) Stick figure model and limb angles

Fig. 5. Joint positions and limb angles in the gait

Apart from the kinematics features, the shape and variation of silhouette are also important characteristics in walking style of human being. Herein, they are discussed in detail in next subsection.

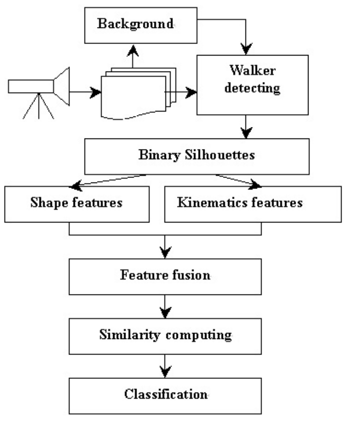
3.2 Shape Features Acquisition

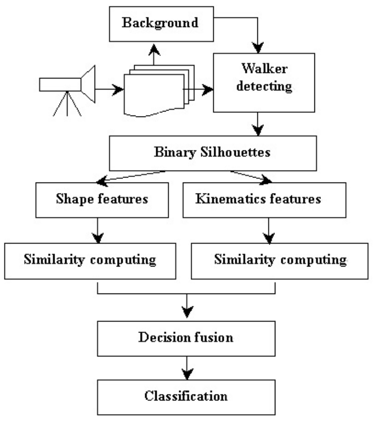
The other two important cues in gait recognition are the width and height of the body, which represent a person's figure well. However, the two features are not always consistent with the human themselves and vary with the camera's focus. As a result, it's not advisable to use them as gait features directly. In this paper, we propose to use the ratio of the silhouette's height to width (H-W ratio) to represent the holistic shape, which is comparatively stable.

However, the division of height by width also eliminates some useful information. For example when one is high and fat and the other one short and slim, then the H-W ratio may be the same or similar. To conquer the above shortcoming, we also count the number of foreground pixels in the silhouette in each frame, together with the H-W ratio, to represent the holistic shape of gait. Likewise, the magnitude normalization has to be done, too.

3.3 Fusion Strategies

The ultimate goal of designing pattern recognition systems is to achieve the best possible classification performance for the task at hand. Fusion of multiple properties of gait is likely to yield tangible benefits. In this paper, two different fusion strategies are used for gait recognition: feature level fusion and decision level fusion. In the feature level fusion, multiple sources of features are combined into a bigger feature as the gait signature (It is noticeable that the features from multiple sources must be normalized into the same interval). And then the pattern training and classification are carried out on the gait signature to recognize different subjects. While in the decision level fusion, multiple sources of features are regarded as independent gait patterns to be trained and to get classification decision data. Fusion is then carried out on these decision data to obtain the final classification result. These decision data, with quite different ranges and distributions, must also be normalized into the same interval before fusion. Herein, two approaches to classifier combination (the Sum and Product rules) are investigated respectively [11]. The flow charts of the two fusion strategies are shown in figure 6.





(a) Feature level fusion strategy (b) Decision level fusion strategy

Fig. 6. Diagram of two different fusion strategies

4 Experimental Results

In this section, we demonstrate the performance of our proposed algorithm on different databases. Our experiments aim to find out how well our method performs with respect to several different variations such as the size of database, speed of walking and etc. We have considered indoor as well as outdoor data for analysis.

4.1 Experimental Data

The video sequences are taken from the following databases, which are used to examine the influence to the size of database, outdoor/indoor and fast/slow walk.

1. The University of California, San Diego (UCSD) database: It's for the outdoor scene. The distance between the camera and subjects is comparatively far. There are 6 subjects, 7 sequences for each subject and 2-3 gait cycles in each sequence. The original images of 320×160 are normalized into 104×104 image templates for our experiments.
2. Carnegie Mellon University (CMU) database: It's for the indoor scene. The distance between the camera and subjects is comparatively small. It has 25 subjects walking at a fast pace and slow pace respectively. There are about 7-8 gait cycles in each sequence. The original images of 640×480 are fitted into 64×64 image template for our experiments.

4.2 Training Phase

Given a gallery with C sequences, a gait feature vector of $(f_1, f_2 ... f_n)$ (n is the dimensionality of the vector) is extracted from each frame of each sequence by using the procedure described in Section 3. The gait feature vector from the j^{th} frame of the i^{th} sequence is denoted by $X_{i,j} = (f_{ij}^1, f_{ij}^2 ... f_{ij}^n)$, $1 \le i \le C$, $1 \le j \le N_i$, where N_i is the number of frames in the sequence.

4.3 Recognition Phase

Let $X_g = \{X_{g,1}, X_{g,2}, \cdots, X_{g,N_g}\}$ be a sequence in the gallery, and $X_p = \{X_{p,1}, X_{p,2}, \cdots, X_{p,N_p}\}$ be an arbitrary one in the probe, where N_g and N_p are the total frame numbers of the two sequences, respectively. Due to the gait's periodicity, we adopt a spatio-temporal similarity measurement based on the gait cycles, which is similar to the method in [2].

Suppose the period length of the gait is N in a sequence. We can partition the whole sequence into $\lfloor N_p / N \rfloor$ subsequences. The k^{th} subsequence is denoted by $X_p(k) = \{X_{p,k+1}, X_{p,k+2}, \cdots, X_{p,k+N}\}$. Then for the subsequence in the probe and some one in the gallery, the distance between them can be calculated as

$$dis_{(X_p(k), X_g)}(l) = \sum_{j=1}^{N} \left\| X_{p,k+j} - X_{g,l+j} \right\| \tag{2}$$

where l is the starting frame in the gallery sequence, from which we compare the two subsequences. The similarity of two whole sequences is defined as

$$Sim(X_p, X_g) = 1 - \frac{1}{K} \sum_{k=1}^{K} \min_l (dis_{(X_p(k), X_g)}(l)) \tag{3}$$

In the experiments, the classification process is carried out using two different classification methods, which are the nearest neighbor classifier (NN) and the K-nearest neighbor classifier (KNN), respectively.

4.4 Experimental Results and Performance Analysis

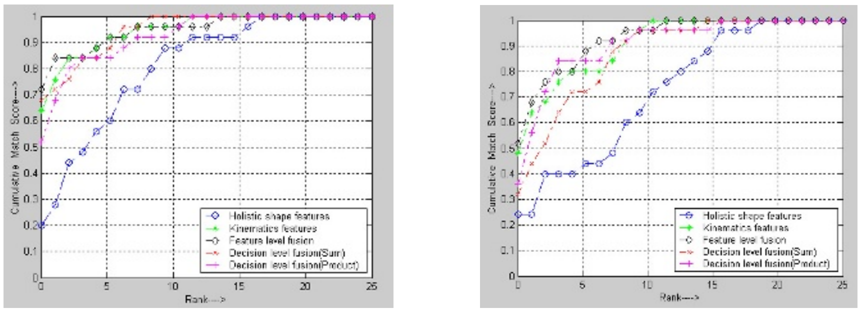
We compute the unbiased estimation of the true classification rate using the leave-one-out cross-validation rule [4] to evaluate the performance of our proposed method. There are 42 sequences in UCSD database, within which we leave one example out as the probe, and the rest as the gallery. The probe sequence is classified according to its similarity with respect to the stored gallery sequences. This process is repeated for 42 times, and the recognition rate is obtained from the number of correctly classified test examples out of the total 42. In the CMU database, there are 8 gait cycles in Fast Walk sequences and 7 cycles in Slow Walk sequences. We can also regard each gait cycle as a sequence, and we execute the leave-one-out process in the same way for the 8 or 7 gait cycles. For the convenience of the later discussion, these experiments are numbered as A-C. With the same experimental circumstance and databases, we carry out the above three experiments by using different feature sets: holistic shape features, kinematics features, feature level fusion and decision level fusion (including the Sum and Product rules), respectively. The correct classification rates (CCR) are summarized in Table 1.

Moreover, in order to evaluate the influence of speed to our method, we complete two other experiments on the CMU database with all the gait cycles. One is training on fast- walk and testing on slow-walk, the other is training on slow-walk and testing on

Table 1. CCR of different algorithm and their comparison

Database		UCSD Database		CMU Database			
Speed of walking		Normal walk		Fast walk		Slow walk	
Experiment		A: Leave-one-out		B: Leave-one-out		C: Leave-one-out	
Classifier		NN	KNN	NN	KNN	NN	KNN
Shape features only (%)		76.19	71.42	83	82.5	85.14	80
Kinematics features only (%)		85.71	90.47	75.5	75.5	76	73.14
Feature level fusion (%)		95.24	97.62	89	90	88.57	85.14
Decision level fusion (%)	Sum	97.62	97.62	92	92	90.86	87.43
	Product	92.86	95.24	93.5	93.5	93.14	88.57

fast-walk. These two experiments are numbered as D and E, respectively. We also use the measure of Cumulative Match Score (CMS) [12] to evaluate the performance of them. By plotting the CMS curves of experiment D and E in Fig.7, we further compare the performance of the above-mentioned algorithms. Note that the horizontal axis of the graph is rank and the vertical axis is the probability $p(i)$ of the identification. Obviously, $p(1)$ is equivalent to the correct classification rates (CCR) of the NN classifier.



(a)CMS of experiment D (train: fast, test: slow) (b)CMS of experiment E (train: slow, test: fast)

Fig. 7. Recognition performance based on CMS

4.5 Discussion

From Table 1, we can see that both shape and kinematics features derived from the walking video can be employed alone for gait recognition, but fusion is indeed a more effective strategy to improve the recognition performance. Basically, the results using fusion strategy are much better than those using holistic shape or kinematics information alone for experiment A, B and C. And the CCR of the experiments across speed test on the CMU database (D and E) has also increased to some extent. The following conclusions are drawn from Table 1 and Fig. 7:

1. The results using kinematics information are somewhat better than those using holistic shape information on UCSD database (see experiment A). The contrary conclusion is drawn on CMU database (see experiment B and C). The reason may be that the distance between the camera and the subject on CMU database is smaller

than that of on UCSD database. Namely, the farther the distance is, the more illegibility the human figure is, so the kinematics information does their work better than shape information, and vice versa.
2. For fusion algorithm, the average CCR of A is better than B and C. This is consistent with the results using kinematics information only. Herein, kinematics information contributes more to fusion results.
3. The average CCR of the decision level fusion using the Sum rule is the highest on UCSD database. However, that of the decision level fusion using the Product rule is the highest on CMU database. On the whole, the identification performance of decision level fusion is better than that of feature level fusion.
4. The identification performance based on the CMS in Figure 7 also demonstrates the effectiveness of fusion. There are somewhat improvements both in the experiment D and E.
5. In experiment D and E for the speed test, the identification performance of holistic shape information is the worst and that of feature level fusion is the best. Furthermore, we can see that the kinematics information contributes more than the holistic shape one in fusion algorithms.

5 Conclusions

In this paper, we propose to represent the holistic shape features by the H-W ratio of body and the pixel number of silhouette. We also propose a feature level fusion approach for gait recognition and compare it with the decision level fusion one. The kinematics features are represented by the joint-angle trajectories of main limbs. Both the holistic shape features and the kinematics ones can be employed alone for gait recognition task, but the fusion is a more effective strategy to improve the recognition performance. The experiments on real sequences of both indoor and outdoor scenes have demonstrated the feasibility of our approach. The identification performance on the larger database needs to be further tested. Future work will concentrate on fusing more sources of gait feature to improve the identification performance, and investigation of more excellent fusion strategies.

Acknowledgment

The work described in the paper was supported by National Natural Science Fund (60403008), Aerospace Innovation Fund, and Aviation Science Fund (03I53065), P. R. China.

References

1. James B. Hayfron-Acquah, Nixon, M. S, John N. Carter .Automatic gait recognition by symmetry analysis. Pattern Recognition Letters, 2003, Volume 24, Issue 13, Pages 2175-2183
2. Zongyi Liu, Sarkar S. Simplest Representation Yet for Gait Recognition: Averaged Silhouette. In Proceedings of the ICPR 17th International Conference on Pattern Recognition, 2004, 4 (23-26): 211 – 214

3. Jonathon Phillips P, Sarkar S, Robledo I, Grother P, Bowyer K. The gait identi.cation challenge problem: Data sets and baseline algorithm. In International Conference on Pattern Recognition, 2002, 385–388
4. Wang L, Tan T., Ning H. and Hu W. Silhouettes Analysis-based Gait Recognition for Human Identification. IEEE Trans. Pattern Anal. and Mach Intell. 2003, 25(12): 1505-1518
5. A. Kale, A.N. Rajagopalan, N. Cuntoor, and V. Kruger. Gait based recognition of humans using continuous HMMs. In Proc. of the International Conference on Face and Gesture Recognition 2002.
6. Cunado, D., Nixon, M. S. and Carter, J. N. Automatic Extraction and Description of Human Gait Models for Recognition Purposes. Computer Vision and Image Understanding, 2003, 90 (1): pp. 1-41
7. Huazhong Ning , Tieniu Tan , Liang Wang and Weiming Hu, Kinematics-based tracking of human walking in monocular video sequences, Image and Vision Computing, 22(5): 429-441 , 2004
8. Tanawongsuwan R, Bobick A. Modelling the Effects of Walking Speed on Appearance-based Gait Recognition. Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004, 2: 783-790
9. Yoo J.-H., Nixon M. S., and Harris C. J.. Extracting Human Gait Signatures by Body Segment Properties. Proceedings of IEEE Southwest Symposium on Image Analysis and Interpretation, 2002, 35-39
10. Liu Yudong, Su Kaina and Ma Li. Gait Recognition Method Based on Body Skeletal Model. Computer Engineering and Applications, 2005, 4(9): 88-92
11. Naresh Cuntoor, Amit Kale and Rama Chellappa. Combining Multiple Evidences for Gait Recognition. Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003, 3: III - 33-6
12. Phillips J, Moon H, Rizvi S, Rause P. The FERET Evaluation Methodology for Face Recognition Algorithms. IEEE Trans Pattern Analysis and Machine Intelligence, 2000, 22(10): 1090-1104

Illumination Normalization for Color Face Images

Faisal R. Al-Osaimi, Mohammed Bennamoun, and Ajmal Mian

The University of Western Australia
35 Stirling Highway, Crawley, WA 6009, Australia
{faisal, bennamou, ajmal}@csse.uwa.edu.au

Abstract. The performance of appearance based face recognition algorithms is adversely affected by illumination variations. Illumination normalization can greatly improve their performance. We present a novel algorithm for illumination normalization of color face images. Face Albedo is estimated from a single color face image and its co-registered 3D image (pointcloud). Unlike existing approaches, our algorithm takes into account both Lambertian and specular reflections as well as attached and cast shadows. Moreover, our algorithm is invariant to facial pose and expression and can effectively handle the case of multiple extended light sources. The approach is based on Phong's lighting model. The parameters of the Phong's model and the number, direction and intensities of the dominant light sources are automatically estimated. Specularities in the face image are used to estimate the directions of the dominant light sources. Next, the 3D face model is ray-casted to find the shadows of every light source. The intensities of the light sources and the parameters of the lighting model are estimated by fitting Phong's model onto the skin data of the face. Experiments were performed on the challenging FRGC v2.0 data and satisfactory results were achieved (the mean fitting error was 6.3% of the maximum color value).

1 Introduction

The acquisition of face biometrics is non-intrusive which makes it suitable for many identification and verification applications. These applications include forensics, security, access control to buildings and services. High recognition accuracy is very crucial for these applications. Unfortunately, variations in lighting conditions significantly degrades the performance of 2D face recognition. The differences amongst images due to identity variations may be obscured by variations caused by illumination.

Many approaches have been proposed to handle the illumination problem. These approaches fall into three main categories namely, (1) illumination insensitive representations, (2) modeling of illumination variations and (3) illumination normalization to a canonical form.

Approaches in the first category are the earliest and the most widely used [3]. In this category, illumination invariant features are generally extracted from the image for better recognition performance. Chen et al. [1] showed that there are no discriminative illumination invariant functions for objects with a Lambertian surface. Therefore, these feature can more appropriately be termed as illumination insensitive as opposed to illumination invariant. Another approach in this category extracts edge maps for recognition [4][5]. Edge maps are compact representations but lack the important information encoded in the intensity shade. Moreover, illumination variations may also create incorrect edges. One may expect edge maps to work better in the case of object recognition compared to face recognition because human faces have a similar structure. Derivatives of gray scale images are also used to overcome the illumination problem [6][7] since they are less sensitive to illumination. However, derivatives are sensitive to noise. Inspired by the fact that the human eye cortex enhances edges, some researchers convolved facial images with Gabor-like filters (which enhance edges) [8][9][10]. However, Adini et al. [2] have shown that edge maps, image derivatives and Gabor-like filters are insufficient to overcome the illumination problem.

The second category i.e. illumination variations modeling, models the face under all possible illuminations. The illumination cone method [11][12] selects three images with constant pose but varying illumination to estimate the set of images under all possible lighting conditions. Ramamoorthi et al. [13] and Basri et al. [14] independently showed that a Lambertian surface under varying illumination can be approximated by the first nine spherical harmonics. Basri et al. [14] performed recognition by comparing the distance between a query face with the nearest images that can be synthesized by the 3D face models and their corresponding albedos under lighting conditions spanned by the first four harmonics [14]. Both [13][14] assume known pose, a Lambertian surface and no cast shadows.

The third category normalizes images to a canonical form by compensating for illumination variations. Histogram equalization [15] and gamma intensity correction [16] fall in this category. However, these methods are global and their output is affected by directional lighting. Shan et al. [17] partitioned a face image into four quarters and used histogram equalization and gamma intensity correction in each quarter to eliminate side-lighting effects. However, due to the complexity of human faces, patterns created by directional lighting cannot be correctly represented by predefined fixed regions. Another approach in this category estimates the direction of a single light source from a generic 3D face and average albedo and relights the input face to a canonical form [18]. Their approach [18] does not cope well with complex lighting conditions caused by multiple light sources with varying intensities. Quotient image relighting is also used to relight face images to a canonical form [19][17]. However, it requires the computation of bootstrap set (the ratio of images in non-canonical forms to a canonical form image) and assumes a similar shape (for a given individual's face) which is not true as the shape of the face significantly changes with expressions. They [19] also claim that their technique could be extended to color images assuming illumination does not affect hue and saturation of an image colors. However, our findings (Section 2.1) show that the saturation of facial skin color varies significantly with changes in illumination.

We present a fully automatic illumination normalization algorithm for color facial images. Our algorithm falls in the third category and unlike other techniques it takes into account the cast shadows, multiple directional light sources (including extended light sources), the effect of illumination on colors and both Lambertian and specular light reflections. In addition, it does not assume any prior knowledge about the facial pose or expressions. Our approach is based on Phong's lighting model which is widely used for rendering in computer graphics. We calculate the face albedo from a single colored face image and its registered 3D model by reversing the image rendering process. First, the number and directions of the dominant lights sources are automatically determined from the specularities in the facial skin. Next, the parameters of Phong's model and the intensities of the light sources are estimated by fitting Phong's model onto the red, green and blue channels of the facial skin. Finally, the parameters of Phong's model, the intensities and directions of light sources and the original facial image and its 3D model are used to calculate the colored face albedo. Experiments were performed on the Face Recognition Grand Challenge (FRGC) v2.0 [27] dataset (9,900 2D and 3D faces) which is challenging in the sense that the faces have major expression variations and are illuminated by varying extended light sources. Our results show that our algorithm can compensate for lighting variations without compromising the local features or effecting the color of the face albedo.

2 Reflective Properties of the Human Face

There are two types of light reflections from surfaces namely Lambertian and specular. A Lambertian surface reflects the incident light equally in all directions but a specular surface reflects the incident light mostly in the mirror reflection direction. In practice, surfaces reflect light with varying proportions of Lambertian and specular components. Shafer et al. [20] show that the sensor response of a red, green or blue color channel R to spectral light $e(\lambda)$ reflected by a surface is given by

$$R = K_L(\hat{n}, \hat{s}) \int f(\lambda)e(\lambda)c_L(\lambda)d\lambda + K_S(\hat{n}, \hat{s}, \hat{v}) \int f(\lambda)e(\lambda)c_S(\lambda)d\lambda \qquad (1)$$

where $f(\lambda)$ is the sensitivity of the color channel (different $f(\lambda)$ for every color channel). c_L and c_S are the albedo and Fresnel reflectance of the surface, respectively. The functions K_L and K_S scale the Lambertian and the specular components. K_S depends on the the normal of the surface $\hat{n}$, light source direction $\hat{s}$ and viewer direction $\hat{v}$ but K_L depends only on $\hat{n}$ and $\hat{s}$.

As stated in Section 1, existing illumination normalization algorithms assume that the human face is a Lambertian surface. In this section, we show that it has a considerable specular component which is responsible for the variations in the color saturation of the facial skin. In fact, illumination causes variations in the saturation even more than the value of the color of face skin. Before proceeding to the details in Section 3, it is important to introduce Phong's lighting model (Section 2.1) and the HSV color space (Section 2.2).

2.1 Phong's Lighting Model

Phong's lighting model is widely used in computer graphics for rendering [21][22]. The model takes into consideration both the Lambertian and specular light reflections. Given a 3D computer model, its albedo, lighting sources and the Phong's model parameters, the image is rendered according to the following equation.

$$\begin{bmatrix} T_R \\ T_G \\ T_B \end{bmatrix} = \begin{bmatrix} A_R D_R \\ A_G D_G \\ A_B D_B \end{bmatrix} + \sum_{i=1}^{n} \left(k_l(\hat{N} \cdot \hat{S}) \begin{bmatrix} A_R L_{R_i} \\ A_G L_{G_i} \\ A_B L_{B_i} \end{bmatrix} + k_s(\hat{V} \cdot \hat{R})^m \begin{bmatrix} F_R L_{R_i} \\ F_G L_{G_i} \\ F_B L_{B_i} \end{bmatrix} \right) \tag{2}$$

where A, D and L_i are the albedo of the surface, ambient diffused light and the intensity of the i-th light source. $\hat{N} \cdot \hat{S}$ is the dot product between surface normal $\hat{N}$ and light source direction $\hat{S}$, and $\hat{V} \cdot \hat{R}$ is the dot product between the viewer direction $\hat{V}$ and the mirror reflection angle of the light source $\hat{R}$. The parameters k_l, k_s and m determine the extent to which a surface is Lambertian. F represents the Fresnel reflectance parameters of a surface i.e. the ratio of red, green and blue components reflected in a specular fashion. Since specularly reflected light from the facial skin has the same color as the incident light, $F = [1 \;\; 1 \;\; 1]^\top$.

2.2 HSV Color Space

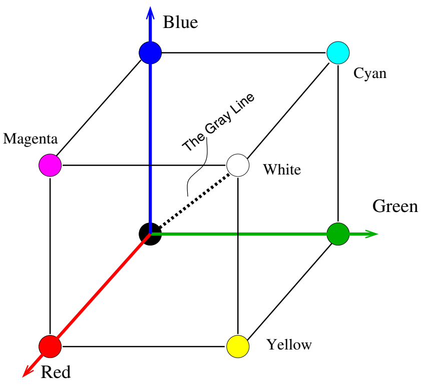
In RGB format, a color is represented by the amount of red, green and blue components it contains. Fig. 1.(a) shows the RGB color cube. If the three color channels are equally balanced, the color is on the gray line (the diagonal connecting the black to the white color). RGB is the most common color format because it is suitable for sensors and display devices. However, with regards to color perception, RGB is not always the best format and sometimes it is desirable to represent colors by their hue, saturation (lightness) and value (brightness) [23]. Fig 1.(b) shows the HSV color space which is an affine transformation of the RGB color cube to a cone. The hue is the angle around the gray line starting from the red color. The saturation axis is perpendicular to the gray line and ranges from 0 to 1. The further the color is from the gray line, the more saturation it has (0 on the gray line and 1 on the sides of the cone). The value of a color (range 0 to 1) is the distance from the black color to the projection of the color on the gray line.

2.3 Variations in Facial Skin Color Due to Illumination

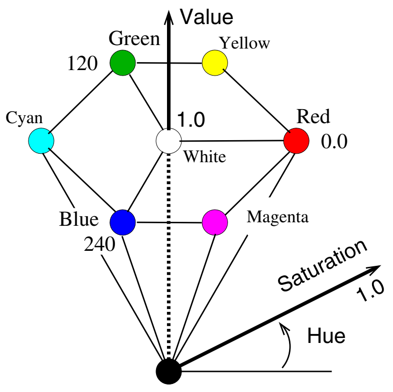
Phong's lighting model is based on the physics of light reflections and is very analogous to the sensor response in Eq. 1. This is the main reason we use this model in our illumination normalization algorithm and analysis. Assuming white light sources (equal R, G and B), Phong's model reduces to

$$\begin{bmatrix} T_R \\ T_G \\ T_B \end{bmatrix} = I_D \begin{bmatrix} A_R \\ A_G \\ A_B \end{bmatrix} + \sum_{i=1}^{n} \left(k_l I_i(\hat{N} \cdot \hat{S}) \begin{bmatrix} A_R \\ A_G \\ A_B \end{bmatrix} + k_s I_i(\hat{V} \cdot \hat{R})^m \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right) \tag{3}$$

where I_i is the intensity of the i-th light source.



(a) RGB color space



(b) HSV color space

Fig. 1. An affine transformation relates HSV and RGB color spaces

Facial skin of the same person generally has a fixed albedo. Let the number of the light sources, their intensities, the geometry of the face, k_l, k_s and m have arbitrary values. Eq. 3, shows that the total color T equals the sum of $n + 1$ vectors (the diffused and Lambertian components) in the direction of the albedo and n vectors (the specular component) in the direction of the white color. Since there are only two independent vectors (the albedo and the white color), the resultant color T will always be in the plane spanned by the albedo and the white color (see Fig. 2.(a)). This implies that T has a constant hue because it does not rotate around the gray line. However, the saturation and value will vary accordingly. As the value increases, the saturation decreases (also see Fig. 2.(d)) because the specular components bring T closer to the gray line.

Scatter plots of hue, saturation and value of a facial skin data (each point represents a pixel) agree with these conclusions (see Fig. 2). Fig. 2.(a) shows that there is very limited variation in the hue of the skin but the value varies significantly. Fig. 2.(b) shows that there is significant variation in the saturation which means that the human face reflects large proportion of the incident light in specular reflection. Results of fitting Phong's lighting model on facial skin data shows that the ratio of the Lambertian to the specular reflection (k_l:k_s) ranges from 1:4 to 1:6.5 (see Section 3.4 and 4).

3 Illumination Normalization Algorithm

In real world environments, illumination conditions are very complex. For example, there are light reflections by objects to the face and extended light sources. However, these can be approximated by a single diffused and a few directional light sources. These unknowns and the parameters of Phong's model are estimated by exploiting the variations in color saturation and value in a facial skin image which are mainly caused by illumination (Section 2). The following subsections give details of our algorithm.

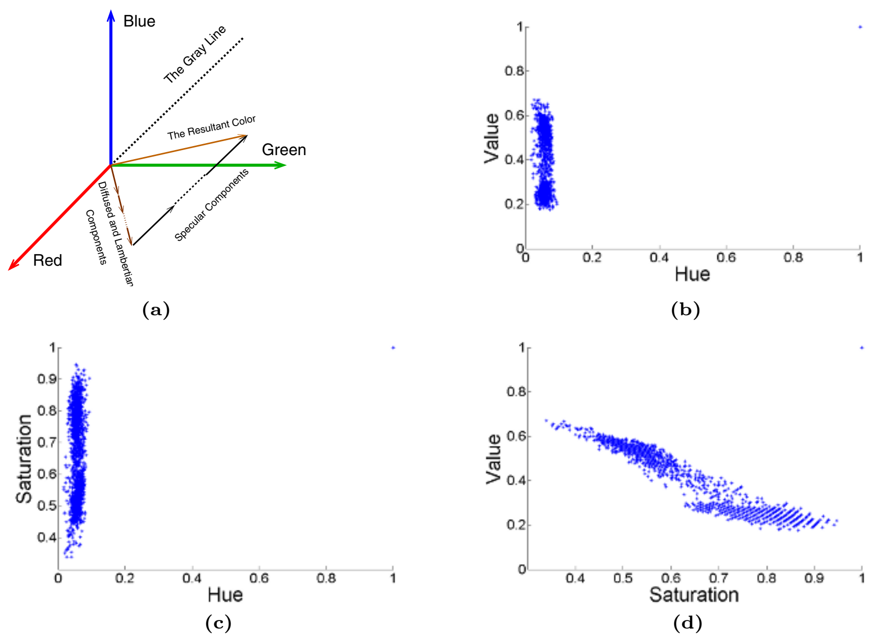


Fig. 2. Illustrations and tests which show that white illumination causes variations in the value and the saturation but not the hue of a facial skin image

3.1 Skin Detection

The skin hue of different human races is very similar [24]. Hue statistics are widely used in skin detection [24][25]. The mean μ_h and standard deviation σ_h of the skin hue are computed from many training images. A pixel p is considered a skin pixel if its hue h_p is within $\pm\ 2.5\mu_h$.

$$S_g = \{p \in S_g | h_p > \mu_h - 2.5\sigma_h \text{ and } h_p < \mu_h + 2.5\sigma_h\} \tag{4}$$

However, the skin pixels set S_g will include pixels from non-skin regions like lips and eye brows because σ_h might be larger than the hue differences between skin and other facial regions of the same face as it is computed from a large set of skin training data. To overcome this problem, we compute the person specific hue statistics μ_{h_s} and σ_{h_s} from S_g and apply another skin detection iteration using μ_{h_s} and σ_{h_s} to produce a more accurate skin pixel set S_s. Since $\sigma_{h_s} \ll \sigma_h$, the pixels which have slight hue differences from the skin are not included in S_s. The pixels which have color values less than a threshold V_{th} are dropped from S_s because they are not reliable. Fig. 3.(a) and 3.(b) show a face image and its detected skin mask, respectively.

3.2 Detection of the Dominant Light Sources

Specularities (highlights) occur at regions at which the mirror reflection direction $\hat{R}$ of a light source is very close to the direction of the viewer (sensor) $\hat{V}$. At these regions $\hat{V} \cdot \hat{R}$ is maximum which means that there are more specular than Lambertian light reflections. In Section 2, we showed that the specular reflection makes the saturation of a color decrease. As specular light reflection from facial skin has more effects on saturation than value, it is more reliable to use saturation for the detection of specularities. A few hundred skin pixels with the minimum saturations are selected. These highlighted pixels may correspond to one or more light sources. For each pixel, the direction of the light source $\hat{S}$ is computed given the viewer direction $\hat{V}$=[0 0 1]^T and the surface normal $\hat{N}$ from the corresponding point in the 3D model as shown in the following equations.

$$\theta_o = \cos^{-1}(\hat{N} \cdot \begin{bmatrix} 0 \; 0 \; 1 \end{bmatrix}^T) \tag{5}$$

$$\hat{P} = \hat{N} \times \begin{bmatrix} 0 \; 0 \; 1 \end{bmatrix}^T \tag{6}$$

$$\begin{bmatrix} S_x \\ S_y \\ S_z \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ N_x & N_y & N_z \\ P_x & P_y & P_z \end{bmatrix}^{-1} \begin{bmatrix} \cos(2\theta_o) \\ \cos(\theta_o) \\ 0 \end{bmatrix} \tag{7}$$

Where θ_o and $\hat{P}$ are the angle and the cross product between $\hat{N}$ and $\hat{V}$, respectively. Azimuth and elevation angles of the light source directions are calculated by changing the $\hat{S}$ vectors from rectangular to spherical coordinates. The azimuth and elevation angles of the highlight pixels which are caused by the same light source will cluster together (see Fig. 3.(d) and 3.(e)). Hierarchal clustering [26] with Cartesian distance as similarity measure is used to cluster light directions into 10 clusters. The clusters which have entries less than 15% of total entries are discarded. The remaining clusters represent the dominant light sources. For each dominant light source, the average direction is computed (center of gravity of the cluster). Wide direction clusters are divided into multiple clusters (i.e. *an extended light source is represented by multiple light sources*).

3.3 Finding Shadows of a Light Source

The human face is self-shadowing. The shadows of an ideal directional light source have a sharp transition from shadow to shine. However, this is not the case in real world illumination. At the edges of a shadow, the light source is partially visible, resulting in continuous shadows. We use fuzzy sets to produce continuous shadows from the direction cluster of a light source. Each cluster is divided into sectors as shown in Fig. 4.(a). The directions in each sector are represented by a direction $\hat{D}_R$. Let $\hat{D} = \begin{bmatrix} \theta \; \phi \end{bmatrix}^\top$ represent an individual direction, $\bar{D}$ is the mean direction of the cluster and n is the number of directions in a sector. $\hat{D}_{R_i}$ of the i-th sector is computed as follows.

$$\hat{M}_{s_i} = \frac{1}{n} \sum_{j=1}^{n} \|\hat{D}_j - \bar{D}\| (\hat{D}_j - \bar{D}) \tag{8}$$

$$\hat{D}_{RMS_i} = \sqrt{\|\hat{M}_{s_i}\|} \frac{\hat{M}_{s_i}}{\|\hat{M}_{s_i}\|} \tag{9}$$

$$\hat{D}_{R_i} = \hat{D}_{RMS_i} + \bar{D} \tag{10}$$

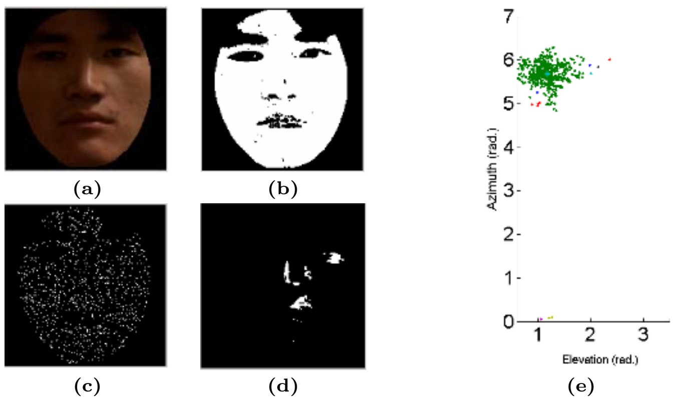


Fig. 3. (a) Original Face image. (b) Skin mask. (c) Randomly selected 1000 representative skin pixels. (d) Detected specularities in facial skin. (e) Direction Clustering: despite that the specular pixels are disconnected spatially in this case they are all pointing roughly to one direction.

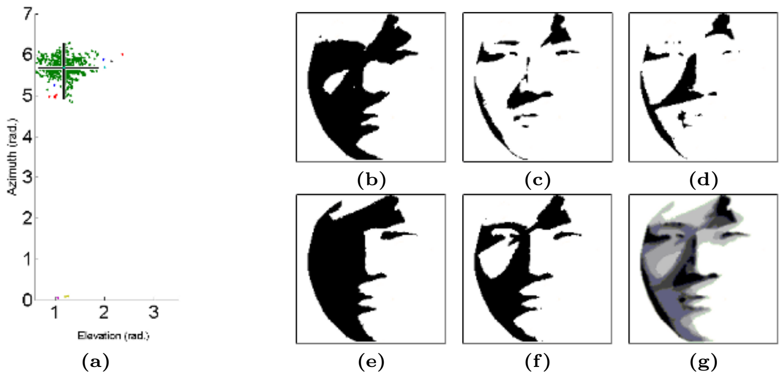


Fig. 4. (a) A direction cluster is divided into 4 sectors. (b),(c),(d),(e) and (f) are the shadows of the sector representing directions and the average direction. (g) the fuzzy shadows of the direction cluster.

The root mean square of the sector directions $\hat{D}_{RMS}$ is used to push $\hat{D}_R$ away from $\bar{D}$. The 3D model is ray-casted by directional lights from $\bar{D}$ and $\hat{D}_R$ of every sector (see Fig. 4) to find the crisp shadow images of these directional light sources (0's for shadows and 1's if shined). The fuzzy shadows membership M of the cluster is calculated by averaging the crisp shadow images and smoothing the resultant image using an average filter. At the center of a shined region, M equals 1 but at edges it gradually changes from 1 to 0.

3.4 Estimation of Light Intensities and Phong's Parameters

At this stage, the number of the dominant light sources, their directions and fuzzy shadows M have been calculated. The intensities of the light sources and Phong's model parameters are estimated by fitting the model (with these variables and $k_l = 1$ as known variables but the free variables are the intensities, an albedo A, k_s and m) on the facial skin. The k_l parameter is kept constant but k_s is variable to avoid redundancy in the free variables. For efficiency, the model is fitted only on 1000 randomly selected skin pixels (see Fig. 3.(c)). Experimental results show that there is no noticeable degradation in performance when using only 1000 skin pixels. However, it considerably cuts the fitting time. Fitting is performed by minimizing the differences between the original skin pixels O and those generated by the model T.

$$\begin{bmatrix} T_{R_i} \\ T_{G_i} \\ T_{B_i} \end{bmatrix} = |I_D| \begin{bmatrix} |A_R| \\ |A_G| \\ |A_B| \end{bmatrix} + \sum_{i=1}^{n} |I_i| \left(M_i(\hat{N} \cdot \hat{S}) \begin{bmatrix} |A_R| \\ |A_G| \\ |A_B| \end{bmatrix} + M_i |k_s| (\hat{V} \cdot \hat{R})^{|m|} \right) \tag{11}$$

$$F = \sum_{i=1}^{1000} \left(|O_{R_i} - T_{R_i}| + |O_{G_i} - T_{G_i}| + |O_{B_i} - T_{B_i}| \right) \tag{12}$$

$$\{A_R, A_G, A_B, I_D, I_1, \cdots, I_n \text{ and } m\} = \arg\min(F) \tag{13}$$

Since the free variables cannot be negative absolute values of the free variables are introduced in Eq. 11 to avoid using a less efficient constrained minimization algorithm. This will make every orthant of the objective function F symmetrical to the positive orthant. Irrespective of which orthant the algorithm converges in, we take the absolute values of the variables. k_s and m do not vary significantly among different faces. By examining the algorithm on many faces, the best results in terms of fitting error and quality of illumination normalization have k_s in the range of 4 to 6.5 and m in the range of 1 to 2. Constraining these two parameters to these ranges gives better results and faster convergence especially when there are many light sources. These constraints are imposed through the objective function to avoid using a constrained minimization algorithm.

$$F = e^{|1.2(k_s - 5.25)|^{1.5}} + e^{|3(m-1.5)|^{1.5}} + \sum_{i=1}^{1000} \left(|O_{R_i} - T_{R_i}| + |O_{G_i} - T_{G_i}| + |O_{B_i} - T_{B_i}| \right) \tag{14}$$

The first two terms have negligible cost if the variables are within the ranges but it grows exponentially outside the ranges.

3.5 Calculation of Face Albedo

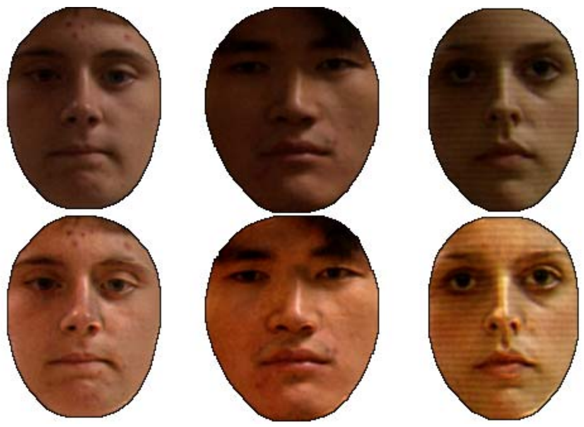
Once the lighting conditions and Phong's parameters are known, face albedo can be computed for every pixel P in the original face image as shown in Eq. 15 (see Fig. 5). The specular components are first subtracted from P. The difference represents the Lambertian components from which the albedo A is computed.

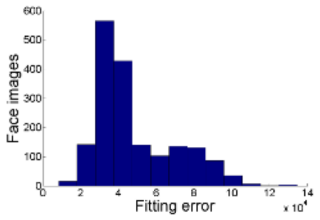
$$\begin{bmatrix} A_{R_i} \\ A_{G_i} \\ A_{B_i} \end{bmatrix} = \begin{bmatrix} (P_{R_i} - \sum_{j=1}^{n} M_{ij} k_s I_j (\hat{V} \cdot \hat{R})^m)/(I_D + \sum_{j=1}^{n} M_{ij} I_j (\hat{N} \cdot \hat{S})) \\ (P_{G_i} - \sum_{j=1}^{n} M_{ij} k_s I_j (\hat{V} \cdot \hat{R})^m)/(I_D + \sum_{j=1}^{n} M_{ij} I_j (\hat{N} \cdot \hat{S})) \\ (P_{B_i} - \sum_{j=1}^{n} M_{ij} k_s I_j (\hat{V} \cdot \hat{R})^m)/(I_D + \sum_{j=1}^{n} M_{ij} I_j (\hat{N} \cdot \hat{S})) \end{bmatrix} \tag{15}$$

4 Results and Discussions

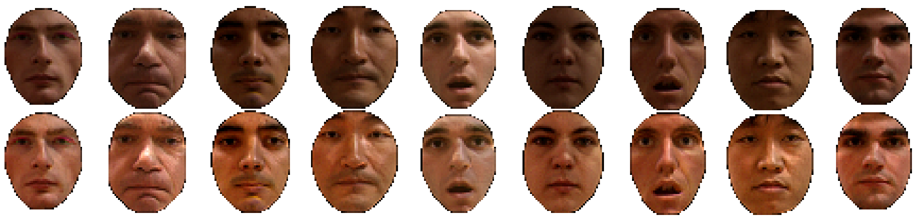
The algorithm was tested on the FRGC v2.0 [27] dataset which contains a large number of co-registered face images and 3D models. Fig. 5 shows some sample images (in color) and their corresponding normalized images produced by our algorithm. A qualitative analysis of the normalized images shows that our algorithm removes substantial amounts of illumination variation effects. The bright regions of a face, which are shined by the dominant light sources, become more similar in color saturation and value to the shadowed regions while preserving the fine details of the face albedo. For example, the red dots in the first face from the left are preserved in the normalized image. Since we only use the dominant light sources, the resulting face albedo (as computed in Eq. 15) looks like a frontal relight of the face more than an ideal albedo which means that there is no need for an additional frontal relight stage.

Fig. 5.(b) shows a histogram of fitting error between the original facial skin and the lighting model (error is defined as in Eq. 12). The average fitting error for the 1000 used skin pixels (each color channel varies in the range 0 to 255) is 4.8×10^4 which gives an average fitting error of $16/pixel.channel$ (about 6.3% of the maximum channel value).





(b)



(a)

Fig. 5. (a) Normalized face images: the original images (top) and the normalized images(bottom)(the figure is better viewed in color) (b) Histogram of fitting error

5 Conclusion

We presented a fully automatic algorithm for the illumination normalization of color face images with minimal assumptions. Our algorithm can accurately estimate the directions and intensities of multiple dominant light sources and the reflective properties of a face. It calculates face albedo by reversing the process of image formation. We stressed on the importance of considering specular reflection of the human face and cast shadows of multiple extended light sources in illumination normalization for facial images. The algorithm was tested on the challenging database of FRGC v2.0 containing complex illumination and facial expression variations. Our results show that the algorithm is robust and accurate.

References

1. H. Chen, P. Belhumuer and D. Jacobs, "In Search of Illumination Invariants," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 254-261, 2000.
2. Y. Adini, Y. Moses and S. Ullman, "Face Recognition: The Problem Of Compensating For Changes In Illumination Direction," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19(7), pp. 721-732, July 1997.
3. David G. Lowe, "Object Recognition from Local Scale-invariant Features," *International Conference on Computer Vision*, pp. 1150-1157, 1999.
4. Y. Gao and M. Leung, "Face Recognition Using Line Edge Map," *IEEE Transactions on Pattern Analysis and Machine Intelligence Pami*, vol. 24(6), pp.764-779, 2002.
5. B. Moghaddam and A. Pentland, "Probabalistic Visual Learning for Object Representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence Pami*, vol. 19(7), pp. 694-710 ,1997.
6. Y. Weiss and , "Deriving Intrinsic Images from Sequences," *In Proc. of Int. Conf. Computer Vision*, 2001.
7. D. Jacobs, P. Belhumeur and R. Basri, "Comparing Images under Variable Illumination," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*,pp. 610-617 ,1998.
8. C. Liu and H. Wechsler , "A Gabor Feature Classifier for Face Recognition ," *In Proc. of Int. Conf. Computer Vision*, 2001.
9. B. Duc, S. Fischer and J. Bigun, "Face authentication with Gabor information on deformable graphs," *IEEE Transactions On Image Processing*, vol. 8(4), pp. 504-516, 1999.
10. M. Lyons, J. Budynek, and S. Akamatsu, "Automatic classification of single facial images," *IEEE Transactions on Pattern Analysis and Machine Intelligence Pami*, vol. 21(12), pp. 1357-1362 , 1999 .
11. A. Georghiades, P. Belhumeur and D. Kriegman , "From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose," *IEEE Transactions on Pattern Analysis and Machine Intelligence Pami*, vol. 23(6), pp. 643-660 , 2001.
12. and , "What is the Set of Images of an Object under all Possible Illumination Conditions?," *International Journal of Computer Vision*, vol. 28(3), pp. 245-260 , 1998.

13. R. Ramamoorthi and P. Hanrahan, "On the relationship between radiance and irradiance: determining the illumination from images of a convex Lambertian object," *Journal- Optical Society of America A*, vol. 18(10), pp. 2448-2459 , 2001.
14. R. Basri and D. Jacobs , "Lambertian Reflectance and Linear Subspaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence Pami* , vol. 25(2), pp. 218-232, 2003.
15. and , "Adaptive Histogram Equalization and its variations," *Computer Vision, Graphics, and Image Processing*, vol. 39(3), pp. 355 - 368 , 1987 .
16. A. Bovik, "Handbook of image and video processing," *Academic Press*,2000.
17. S. Shan, W. Gao, B. Cao and D.Zhao, "Illumination Normalization for Robust Face Recognition Against Varying Lighting Conditions," *IEEE Workshop on AMFG*,2003.
18. C. Lee and B. Moghaddam, "A Practical Face Relighting Method for Directional Light Normalization," *AMFG*, 2005.
19. Riklin-Raviv and A. Shashua, "The Quotient Image: Class-Based Re-Rendering and Recognition with Varying Illuminations," *IEEE Transactions on Pattern Analysis and Machine Intelligence Pami*, vol. 23(2), pp.129-139 , 2001.
20. S. Shafer, "Using Color to Separate Reflection Components," *COLOR Research and Applications* , vol. 10(4), pp. 210-218, 1985.
21. B. Phong, "Illumination for computer generated images," *Comm. ACM* , vol. 18(6), pp. 311-317, 1975.
22. J. Foley, A. Dam, S. Feiner and J. Hughes, "Computer Graphics: Principles and Practice," *Addison-Wesley* , 1990.
23. J. Wang, W. Yang and R. Acharya, "Color clustering techniques for color-content-based image retrieval from image databases," *IEEE International Conference on Multimedia Computing and Systems* ,pp. 442 - 449 , 1997.
24. K. Sobottka and I. Pitas, "A Novel Method for Automatic Face Segmentation, Facial Feature Extraction and Tracking," *Signal Processing: Image Comm.*, vol. 12(3), pp. 263-281, 1998.
25. S. Phung, A. Bouzerdoum and D. Chai , "Skin segmentation using color pixel classification: analysis and comparison," *IEEE Transactions on Pattern Analysis and Machine Intelligence Pami*, vol. 27(1), pp. 148- 154, 2005.
26. A. Jain, M. Murty, and P.Flynn, "Data Clustering: A Review," *Computing Surveys*, vol. 31(3), pp. 264-323, 1999.
27. P. Phillips, P. Flynn, T. Scruggs, K. Bowyer, J. Chang, K. Hoffman, J. Marques, J. Min and W. Worek, "Overview of the Face Recognition Grand Challenge," *IEEE Computer Vision and Pattern Recognition*, vol. 1, pp. 947 - 954 , 2005.

Real-Time Detection of Out-of-Plane Objects in Stereo Vision

Weiguang Guan and Patricia Monger

Dept. of Research and High Performance Computing
McMaster University, Hamilton, Ontario, Canada, L8S 4L7
{guanw, monger}@mcmaster.ca

Abstract. This paper proposes an automatic approach to detecting objects appearing in front of planar background. A planar homography is estimated with high accuracy in an off-line initialization phase. Given a pair of binocular images, we apply the estimated homography to one of the images, and then compute a similarity map between the transformed image and the other. Normalized cross-correlation is used in the computation of the similarity map to measure the similarity between neighborhoods of overlapping pixels. Normalized cross-correlation measure is superior to absolute difference in alleviating the influence of image noise and small mis-alignment caused by imperfect homography estimation. The similarity map with pixel intensities ranging between 0 and 1 leads to an easy detection of out-of-plane objects because the values of pixels corresponding to planar background are close to 1. Tracking could be incorporated with our out-of-plane object detection method to further improve robustness in live video applications. This approach has been used in tracking people and demonstrated reliable performance.

1 Introduction

Detection of moving objects has gained a lot of importance in the last few years. It is an indispensable step leading towards understanding moving objects, e. g., identification of people, recognition of their behaviors and/or gestures. A typical method of object detection is background subtraction [4], which detects foreground objects by comparing live images to the statistic model of a static background. However, Background subtraction is not robust against illumination changes, especially when such changes are abrupt.

We often see planar structures in a 3D scene, examples include floors, walls, screens, etc. Real-time detection of objects that are out of planar background has many potential applications, such as detecting and tracking people on floors [3], locating obstacles in autonomous navigation systems [2], and so on. This paper introduces a stereo vision based method for detecting objects in front of planar background which may vary photometrically during detection process.

Plane has been extensively studied and used in computer vision due to its mathematical simplicity. Transformation between a plane in a 3D world and a camera's image plane is defined by a projective homography, a 3×3 matrix up to a scale, under the assumption that the camera observes the pin-hole model. When more than one camera is used, the transformation between any pair of the image planes forms a unique homography with respect to a plane in 3D scene.

Our method utilizes the homography between two stereo images, and avoids the error-prone correspondence problem. First, homography is estimated as accurately as possible. During real-time detection, one image is super-imposed onto the other by applying the estimated homography, followed by calculation of a similairty map. The similarity map with pixel values ranging from 0 to 1 can be used in out-of-plane object detection because pixels corresponding to planar background have values close to 1. The foreground pixels, on the contrary, have lower values, which depend on the richness of local textures.

Normalized cross-correlation is used as similarity measure. It has an important characteristic, scalability, which allows us to adjust the size of local areas (or neighborhoods) to be taken into similarity calculation. The larger the scale is, the more robust against noise and mis-alignment but less sensitive to out-of-plane objects the similarity measure becomes.

The rest of the paper is organized as follows: We introduce our homography estimation method in section 2. It can be considered as an adaptation of the original direct linear transform (DLT) for a group of horizontal and vertical line correspondences. In section 3, we describe how to compute similarity map together with strategy to accelerate the computation. Segmentation of foreground objects out of similarity map is briefly discussed in section 4. Experimental results are presented and analyzed in section 5. Finally, the paper is concluded in section 6.

2 Homography Estimation

Suppose a plane P is imaged by two cameras at different angles. Let I and I' be images of the plane from the two cameras. If the cameras satisfy the pin-hole model, then the I-to-I' image transformation can be described by a planar homography, a non-singular 3×3 projective matrix.

Given a pair of homogeneous coordinates of pixels $\boldsymbol{p} = [x, y, 1]^T$ in I and $\boldsymbol{p'} = [x', y', 1]^T$ in I' that correspond to the same point on plane P in a 3D space, there exist a planar homography $\mathbf{H}_P$ in regard to plane P

$$\lambda \boldsymbol{p'} = \mathbf{H}_P \, \boldsymbol{p} \tag{1}$$

where

$$\mathbf{H}_P = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

and λ is a scaling factor. We can determine $\mathbf{H}_P$ up to a scale, which means $\mathbf{H}_P$ has 8 degrees of freedom.

In projective geometry, there exists duality between line and point. If a point $\boldsymbol{p} = [wx, wy, w]^T$ is on a line $\boldsymbol{l} = [a, b, c]^T$, then

$$\boldsymbol{p}^T \boldsymbol{l} = \boldsymbol{l}^T \boldsymbol{p} = w \cdot (ax + by + c) = 0$$

Because of the duality, we can estimate homography $\mathbf{H}_P$ from line correspondences instead of point correspondences as below

$$\alpha \boldsymbol{l'} = {\mathbf{H}_P}^{-T} \boldsymbol{l} \tag{2}$$

where

$${\mathbf{H}_P}^{-T} = ({\mathbf{H}_P}^{T})^{-1} = ({\mathbf{H}_P}^{-1})^{T}$$

Formula (2) can be re-written as

$$\beta \boldsymbol{l} = {\mathbf{H}_P}^{T} \boldsymbol{l}' \tag{3}$$

where $\beta = 1/\alpha$. Equation (3) is the counterpart of (1) as result of the duality, and will be used in our homography estimation.

Since one pair of point (or line) correspondences gives rise to three linear equations in h_{11}, h_{12}, ... , h_{33}, of which only two are linearly independent, so we need at least four pairs of correspondences of non-collinear points (or non-concurrent lines) to determine the underlying homography. If there are more than four pairs of correspondences available we can obtain a least-squares solution.

We use a black-and-while checkerboard pattern as reference to calculate homography. We synthesize such a pattern and project it on a wall with an LCD projector. Horizontal and vertical lines of the pattern are extracted in both images and paired up. We choose lines rather than points as features to form correspondences as corner detection is sensitive to image noises while line detection is more robust and accurate.

2.1 Line Detection

Our stereo vision system is set up in such a way that a checkerboard pattern is projected onto a planar screen and is completely visible from both of the cameras. Horizontal and vertical lines of the pattern are detected using Hough transform. Since the horizontal and vertical lines of the checkerboard pattern are roughly horizontal and vertical in both images so that we can detect them separately. Below we will only introduce how to detect horizontal lines, vertical lines are detected in the same manner except for the direction.

We first apply horizontal Prewitt edge detector to image. Most directional edge detectors are acceptable for edge detection. Then, histogram analysis of the edge map must be performed in order to determine a proper thresheld to extract these horizontal edges.

Given an edge point $[x, y]$, Hough transform

$$\rho = x \cos\theta + y \sin\theta \tag{4}$$

translates it into a curve in the ρ-θ parameter space. Since the lines to be detected are roughly horizontal, we can narrow θ down to range $[90^o - \delta\theta,\ 90^o + \delta\theta]$ ($\delta\theta = 25^o$ in our experiment) in parameter space instead of full angle range. The range of ρ can be truncated into $[-h/2,\ h/2]$ (h is image height). This implies that we can have higher resolutions for both ρ and θ at the same cost of memory and computation.

As the checkerboard pattern is known, we need to find exactly as many horizontal/vertical lines as existing in the pattern. Detected horizotal lines are sorted from bottom to top while vertical lines are sorted from left to right based on their ρ values. This ordering assures we can have correct line correspondences between left and right images so that RANSAC [1] is not necessary.

2.2 Adaptation of Normalized DLT

Planar homography is defined by a non-singular 3×3 matrix up to a scale, as shown in equations (1) and (3). The traditional method for estimating homography is direct linear transformation (DLT) [6]. We will adapt the original DLT so that line correspondences, instead of point correspondences, are used in homography estimation.

We can get rid of the scaling factor β by taking cross-product with $\boldsymbol{l}$ on both sides of equation (3):

$$\boldsymbol{l} \times (\mathbf{H}_P{}^T \boldsymbol{l}') = \mathbf{0} \tag{5}$$

Let's denote $\mathbf{H}_P = [\boldsymbol{h}_1, \boldsymbol{h}_2, \boldsymbol{h}_3]$, where $\boldsymbol{h}_j$ is the j-th column of $\mathbf{H}_P$. Then

$$\mathbf{H}_P{}^T \boldsymbol{l}' = [\boldsymbol{h}_1{}^T \boldsymbol{l}', \boldsymbol{h}_2{}^T \boldsymbol{l}', \boldsymbol{h}_3{}^T \boldsymbol{l}']^T$$

and equation (5) can be re-written as

$$\begin{bmatrix} b\boldsymbol{h}_3{}^T \boldsymbol{l}' - c\boldsymbol{h}_2{}^T \boldsymbol{l}' \\ c\boldsymbol{h}_1{}^T \boldsymbol{l}' - a\boldsymbol{h}_3{}^T \boldsymbol{l}' \\ a\boldsymbol{h}_2{}^T \boldsymbol{l}' - b\boldsymbol{h}_1{}^T \boldsymbol{l}' \end{bmatrix} = \mathbf{0} \tag{6}$$

or in the form of matrix equation:

$$\mathbf{M}\boldsymbol{h} = \mathbf{0} \tag{7}$$

where

$$\mathbf{M} = \begin{bmatrix} \mathbf{0}^T & -c\boldsymbol{l}'^T & b\boldsymbol{l}'^T \\ c\boldsymbol{l}'^T & \mathbf{0}^T & -a\boldsymbol{l}'^T \\ -b\boldsymbol{l}'^T & a\boldsymbol{l}'^T & \mathbf{0}^T \end{bmatrix} \tag{8}$$

and

$$\boldsymbol{h} = \begin{bmatrix} \boldsymbol{h}_1 \\ \boldsymbol{h}_2 \\ \boldsymbol{h}_3 \end{bmatrix}$$

Like a pair of point correspondences, a pair of line correspondences gives rise to three linear equations in h_{11}, h_{12}, ..., h_{33}, of which only two are linearly independent. However, unlike point correspondences, things become a little more complicated with line correspondences since one of the following two cases could happen (1) $a = c = 0$; or (2) $b = c = 0$. This means that there is not a general rule to discard an equation from (7). We do not have the dilemma as the lines extracted from checkerboard are group into horizontal and vertical ones. We can discard the second equation in (7) for horizontal lines, and first equation for vertical lines.

Given a set of line correspondences $\boldsymbol{l_i} \leftrightarrow \boldsymbol{l}'_i$, $i = 1, 2, ..., N$ $(N \geq 4)$ found in the previous section, we are able to stack N 2×9 matrices into a $2N \times 9$ matrix $\mathbf{A}$ so that

$$\mathbf{A}\boldsymbol{h} = \mathbf{0} \tag{9}$$

where

$$\mathbf{A} = \begin{bmatrix} \hat{\mathbf{M}}_1 \\ \hat{\mathbf{M}}_2 \\ \cdots \\ \hat{\mathbf{M}}_N \end{bmatrix}$$

where $\hat{\mathbf{M}}_i$ represents the coefficient matrix of the two linearly independent equations, obtained by discarding one equation from (7).

The rank of $\mathbf{A}$ is 8 in the pure mathematical sense. In practice, both image noise and discretization influence the accuracy of line detection so that the rank is 9. Direct solution will suffer from instability due to the near-singularity of the problem. We need to impose an extra constraint to obtain a numerically stable solution, $||\boldsymbol{h}|| = 1$, and translate the problem into a minimization problem, i.e., finding $\boldsymbol{h}$ to minimize

$$||\mathbf{A}\boldsymbol{h}|| \text{ subject to constraint: } ||\boldsymbol{h}|| = 1$$

or

$$\frac{||\mathbf{A}\boldsymbol{h}||}{||\boldsymbol{h}||}$$

The non-zero solution is the (unit) eigenvector of $\mathbf{A}^T\mathbf{A}$ with the least eigenvalue. Equivalently, the solution is the right singular vector associated with the smallest singular value of $\mathbf{A}$, which can be solved by singular value decomposition (SVD) of $\mathbf{A}$:

$$\mathbf{A} = \mathbf{UDV} \tag{10}$$

where $\mathbf{D}$ is a diagonal matrix with values sorted in descending order down the diagonal. The last column of $\mathbf{V}$ corresponding the smallest singular value in $\mathbf{D}$ is the solution for $\boldsymbol{h}$. This is known as direct linear transformation (DLT).

The original DLT has the reputation of poor performance. Hartley and Zisserman [6] concluded that normalization of points could greatly improve the accuracy of estimated homography. Such normalization has the effect of equalizing the error caused by each individual correspondence and therefore making the solution more stable.

It is not trivial to adapt normalized DLT from point correspondence based to be line correspondence based although point and line are dual elements in projective geometry. For point correspndences, one can translate points so that their centroid coincide with the origin, followed by scaling so that the mean distance to the origin is $\sqrt{2}$. Unfortunately, there is no counterpart for line correspondences.

We normalize the projective coordinates of each pair of lines independently so that $||\boldsymbol{l}_i|| = 1$ and $||\boldsymbol{l'}_i|| = 1$. DLT yields a least-squares solution when $N > 4$, in a sense of minimizing mean algebraic distance. Our normalization, in fact, equalizes the algebraic error from each pair of line correspondence, achieving effect similar to the normalization proposed by Hartley [6]. Unlike Hartley's normalization, we don't need to transform the resultant homography to recover the homography that corresponds to non-normalized coordinates.

3 Similarity Map

Different points on the surfaces of 3D objects are likely to have different colors/intensities because (1) they might be made of different materials; and/or (2) they might have different surface normals; and/or (3) they might be illuminated differently. It is rare to see two randomly-selected points have the same (or close) color/intensity in a texture-rich scene. When a 3D scene is imaged from two different points of view, pixels of two distinct physical points are unlikely to be of the same color/intensity while pixels of the same point in 3D will have the same or very close color/intensity.

Based on this observation, we have developed the concept of our similarity map. For a pair of binocular images of a scene with planar background, we compute a 2D array (or image) of the same size as that of the binocular images, called similarity map, in which intensities, ranging from 0 to 1, reflect the likelihood of pixels being planar background.

Our similarity map is calculated from a pair of gray-scale images. If color images are available, one can either convert them into gray-scale images before computing similarity map, or compute one similarity map from each of the R, G, B channels and somehow combine the three similarity maps afterwards. The latter, which takes advantage of color information, generally produce better results.

Given a pair of stereo images I and I', one image I is super-imposed upon the other I' by transforming image I with the estimated homography $\mathbf{H}_P$ that is in regard to plane P in a 3D scene. There three possible cases of a overlapped pixel pair:

1. both $I_t[i,j]$ and $I'[i,j]$ are background;
2. both $I_t[i,j]$ and $I'[i,j]$ are foreground;
3. one is foreground and the other is background.

where I_t is transformed I. You can see that $I_t[i,j]$ and $I'[i,j]$ correspond to the same point (on the plane P) only in the first case. In other words, if $I_t[i,j] \approx I'[i,j]$, they are more likely to be background than foreground. In the two remaining cases, $I_t[i,j]$ and $I'[i,j]$ correspond to different 3D points and are therefore more likely to be different.

A simple and straightforward way to distinguish background from foreground is image difference, which is known to be senstitive to noise. It is desirable to have similarity measure between two pixels that is based on their neighborhoods rather than themselves alone. We propose a scalable similarity measure to overcome the influences of image noise and possible mis-alignment caused by imperfect registration in homography estimation. A similarity map $S^{(k)}_{I_1 \leftrightarrow I_2}$ between images I_1 and I_2 with kernel size k at pixel $[i, j]$ is defined as normalized cross-correlation between two neighborhoods centered at the pixel:

$$S^{(k)}_{I_1 \leftrightarrow I_2}[i,j] = \frac{R^{(k)}_{I_1 I_2}[i,j]}{\sqrt{R^{(k)}_{I_1 I_1}[i,j] \cdot R^{(k)}_{I_2 I_2}[i,j]}} \tag{11}$$

where

$$R^{(k)}_{I_1 I_2}[i,j] = \sum_{u=-k/2}^{k/2} \sum_{v=-k/2}^{k/2} I_1[i+u, j+v] \cdot I_2[i+u, j+v]$$

and $I_1 = I'$ and $I_2 = I_t$. Obviously, $0 \leq S^{(k)}_{I \leftrightarrow I'}[i,j] \leq 1$. Parameter k allows us to tune up the similarity measure to compromise between noise-resistance and sensitivity to difference.

Direct calculation of similarity map sized of $m \times n$ could be very time-consuming ($O(k^2 mn)$), which is not acceptable in a real-time application. We have figured out an efficient way to compute our similarity map, inspired by Viola and Jones in their work [5].

Prior to computing similarity map, we compute integral images and keep them in memory. The integral image $J_{I_1 I_2}$ is defined as:

$$J_{I_1 I_2}[i,j] = \sum_{u=1}^{i} \sum_{v=1}^{j} I_1[u,v] \cdot I_2[u,v]$$

The computation of an integral image can be implemented in a recursive way:

1. $J_{I_1 I_2}[1,1] = I_1[1,1] \cdot I_2[1,1]$;
2. $J_{I_1 I_2}[i,j] = I_1[i,j] \cdot I_2[i,j] + J_{I_1 I_2}[i, j-1] + J_{I_1 I_2}[i-1, j] - J_{I_1 I_2}[i-1, j-1]$.

which can be completed in $O(mn)$. Then $R^{(k)}_{I_1 I_2}$ can be immediately retrieved from integral image $J_{I_1 I_2}$ in the following way:

$$\begin{aligned} R^{(k)}_{I_1 I_2}[i,j] = J_{I_1 I_2}[i,j] + J_{I_1 I_2}[i-k, j-k] - \\ J_{I_1 I_2}[i, j-k] - J_{I_1 I_2}[i-k, j] \end{aligned}$$

$R^{(k)}_{I_1 I_1}$ and $R^{(k)}_{I_2 I_2}$ can be acquired in a similar way from $J_{I_1 I_1}$ and $J_{I_2 I_2}$ respectively.

4 Detection of Out-of-Plane Objects

In most cases, it is sufficient to segment the foreground objects by simply thresholding the similarity map. More sophisticated methods, such as morphological

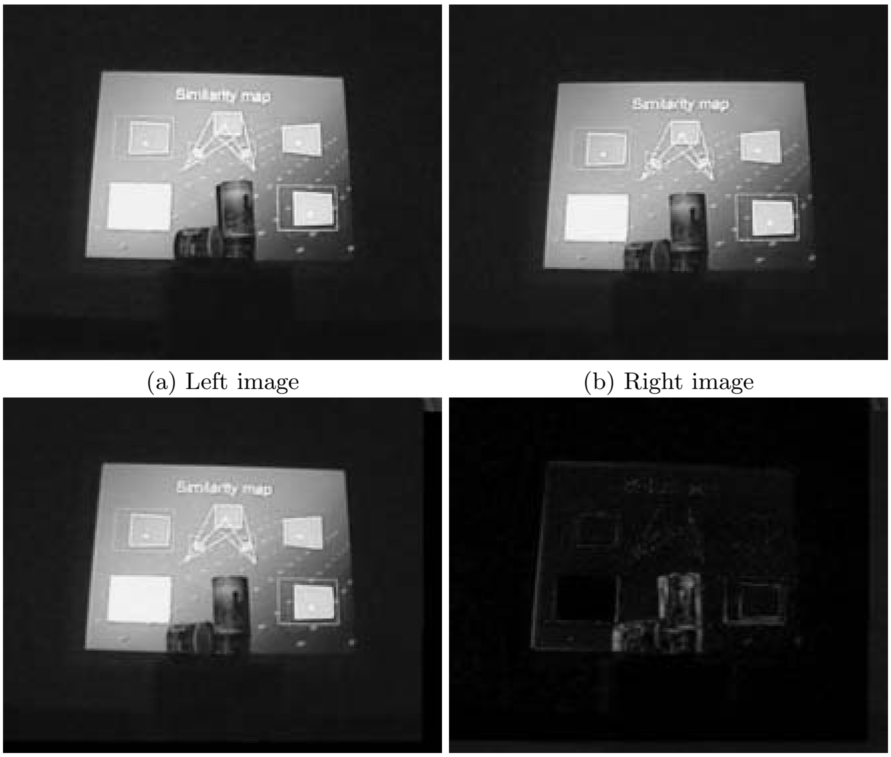

Similarity map
(a) Left image
(b) Right image
(c) Transformed right image
(d) Difference between (a) and (c)

Fig. 1. Stereo, transformed and absolute difference images

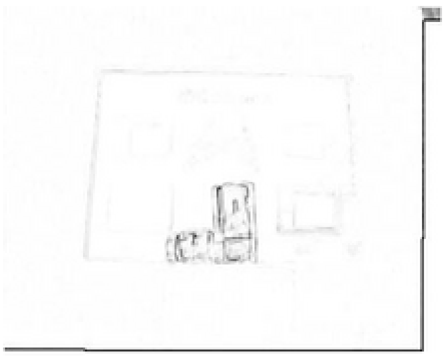
operations, shape analysis based on domain-specific knowledge could greatly improve the segmentation results.

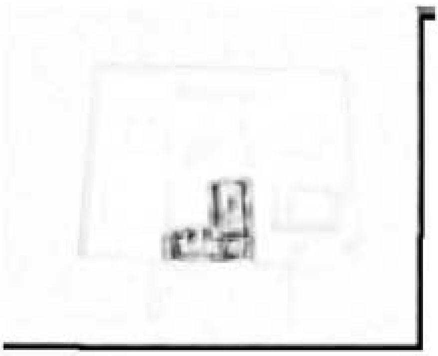
There is a "double-edge" phenomenon in the similarity map, which makes the foreground objects appear larger than they really are. The phenomenon is less visible if the two cameras stay close to each other and the foreground objects are not very close to cameras.

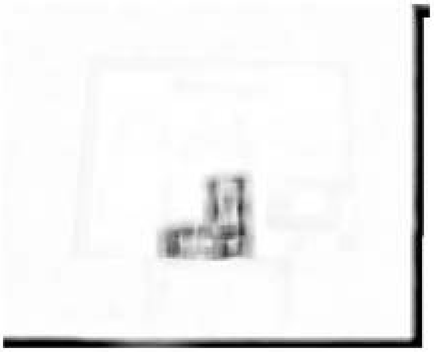
5 Experiments

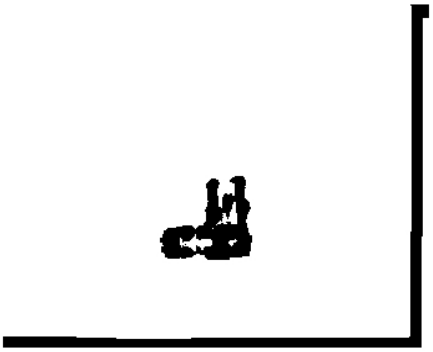
We have tested our method on a PC computer with 1.7GHz CPU. Images were captured through two USB cameras at resolution of 352×288. The original color images were first converted to 8-bit gray-scale images before being processed. The detection system can work at more than 20 frames per second.

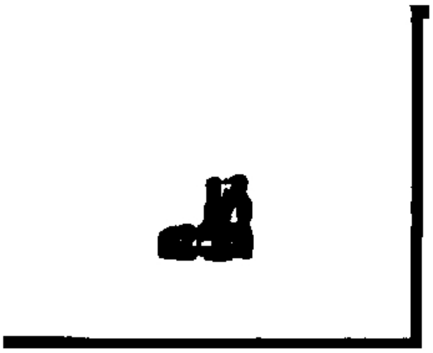
In the initialization stage, we synthesized a checkerboard pattern and projected it on a wall, which was then captured by two cameras. Five horizontal and five vertical lines in the pattern were extracted and paired up for homography estimation, as discussed in section 2. In fig. 1, (a) and (b) are two images of a projected slide with three cans in front of the wall. We super-imposed the


(a) Similarity map with kernel size = 3


(b) Similarity map with kernel size = 7


(c) Similarity map with kernel size = 11


(d) Threshold image of (c) at T = 200


(e) Threshold image of (c) at T = 220

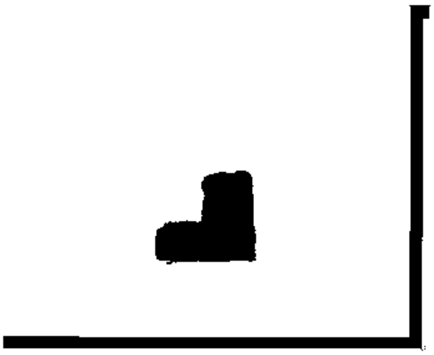

(f) Threshold image of (c) at T = 240

Fig. 2. Similarity maps at different scales

right image over the left image by warpping the right image, and compute the difference between the warpped image and the left image is shown in (d).

Our scalable similarity maps (scaled to range 0-255) were calculated at three different scales in Fig. 2 (a)-(c). You can see that larger scales depress more the impact of mis-alignment while at the same time blur the foreground objects. Objects can be segmented out by thresholding one of the similarity maps, and wide range of threshold values can achieve decent segmentation of the foreground objects as shown in (d)-(f).

6 Conclusions

In this paper we describe a fully automatic, real-time approach to detecting objects in front of a planar background. Our approach is superior to background subtraction as it works when the planar background has dynamically varying light reflection on its surface. The the variation in light reflection may be caused by prejected video or variation in environment illumination.

The method has been successfully used in detection of people who are giving presentations in front of a screen. Good performance results from the combination of our homography estimation and similarity measure. The method can be extended to the case where background is composed of multiple planes, and we need to estimate multiple homographies.

Viewing of the same point on an object surface from different angles may result in variation in perception of color due to the fact that the specular component in the reflected light vary in view point according to Phong's illumination model. This may break the assumption that $I_t[i,j] \approx I'[i,j]$ for planar background. One solution to this problem is bringing two cameras as close as possible, or decreasing parallax in stereo vision terminology. However, you need to keep certain amount of parallax to reveal out-of-plane objects. There is a trade-off in stereo vision in regard to camera setup, which should be determined individually for a particular application. Another solution is photometrical calbration in the initialization stage if light sources do not change.

Acknowledgements. We would like to thank our colleagues in RHPCS for their valuable suggestions and supports.

References

1. Vincent, E. and Laganire, R., "Detecting Planar Homographies in an Image Pair", IEEE Symp. Image and Signal Processing and Analysis., Croatia, pp 182–187, 2001
2. Williamson, T. and Thorpe C., "A Specialized Multibaseline Stereo Technique for Obstacle Detection", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, June, 1998
3. Fleuret, F., Lengagne, R., Fua, P., "Fixed Point Probability Field for Complex Occlusion Handling", ICCV, 694-700, 2005
4. Ohta, N., "A Statistical Approach to Background Subtraction for Surveillance Systems", ICCV, (II: 481-486), 2001.
5. Viola, P., Jones, M. J., "Robust real-time object detection", Technical Report CRL 20001/01, Cambridge Research Laboratory, 2001.
6. Hartley, R. I. and Zisserman, A., "Multiple View Geometry in Computer Vision", Cambridge University Press, 2000.

Stereo Imaging with Uncalibrated Camera

Xiaokun Li[1], Chiman Kwan[1], and Baoxin Li[2]

[1] Signal/Image Processing, Intelligent Automation Inc., MD 20855, USA
[2] Computer Science and Eng. Dept., Arizona State University, AZ 85287, USA

Abstract. 3D images provide more information to human than their 2D counterparts and have many applications in entertainment, scientific data visualization, etc. The ability to generate accurate 3D dynamic scene and 3D movie from uncalibrated cameras is a challenge. We propose a systematic approach to stereo image/video generation. With our proposed approach, a realistic 3D scene can be created via either a single uncalibrated moving camera or two synchronized cameras. 3D video can also be generated through multiple synchronized video streams. Our approach first uses a Gabor filter bank to extract image features. Second, we develop an improved Elastic Graph Matching method to perform reliable image registration from multi-view images or video frames. Third, a fast and efficient image rectification method based on multi-view geometry is presented to create stereo image pairs. Extensive tests using real images collected from widely separated cameras were performed to test our proposed approach.

1 Introduction

In entertainment, gaming, and TV programs, one of the major steps towards natural and easy understanding and perception of image/video is 3D effects. Besides these applications, 3D image/video techniques are needed in many other civilian applications, i.e. medical operation, microscope, scientific data display, CAD/CAM, and surveillance systems. These technologies can also be used directly in target detection and recognition, precise strike, fly guidance, and some other military related applications. Actually, computer-based stereo vision system has been studied for many years, in which the major task is the estimation of 3D depth of the physical scene from a pair of cameras emulating the left and the right eyes in a human visual system. The fundamentals of depth estimation from a pair of stereo cameras (a stereo rig) can be illustrated in the following figure (Fig. 1). But, there are many limitations of conventional stereo vision systems, including the requirements of two identical cameras, narrow baseline, fixed parameter setting and position, limited field of view, and only suitable for short-range scene, etc. All of the above issues seriously limit the usage of conventional stereo vision systems in many areas. In real life, the cameras/sensors used for data acquisition are often nonstationary and located at different viewpoints which have wide distance between each other. Furthermore, the parameter settings of camera are usually unknown and unfixed during data acquisition.

To generate a stereo (3D) image through a single moving camera or multiple synchronized cameras far from each other is a challenging problem. The main reason comes from the fact that, in general, the cameras are not calibrated, thus simply using standard stereoscopic vision algorithms may not generate correct stereo images. Hence, advanced algorithms that can handle uncalibrated cameras are needed. In recent years, many research efforts have been made. Fusiello *et al.* (UK and Italy) [1] presented a compact and efficient algorithm to generate stereo image via image rectification. But, their approach assumes that the stereo rig is calibrated, which means the camera's intrinsic parameters such as focal length, aspect ratio, and the relative position to each other are already precisely known or calculated. Unfortunately, as mentioned earlier, the camera's parameters are hard to know and the relative position between cameras are difficult to obtain or calibrate in practice. Loop and Zhang at Microsoft Research [9] developed one method to construct stereo image with uncalibrated cameras. Their method is mainly designed for stereo matching and the residual distortion may prevent stereoscopic visualization. Hartley (GE Research) & Zisserman (U. of Oxford) [7], [8] proposed a novel method for stereo generation from uncalibrated cameras, which is the most advanced method in the literature based on our knowledge. One important advantage of their method is that the method is robust to many situations. However, one major limitation of the method is the quality of the stereo images can not be guaranteed. Undetermined image deformations such as shrink or distortion often occur in the stereo images.

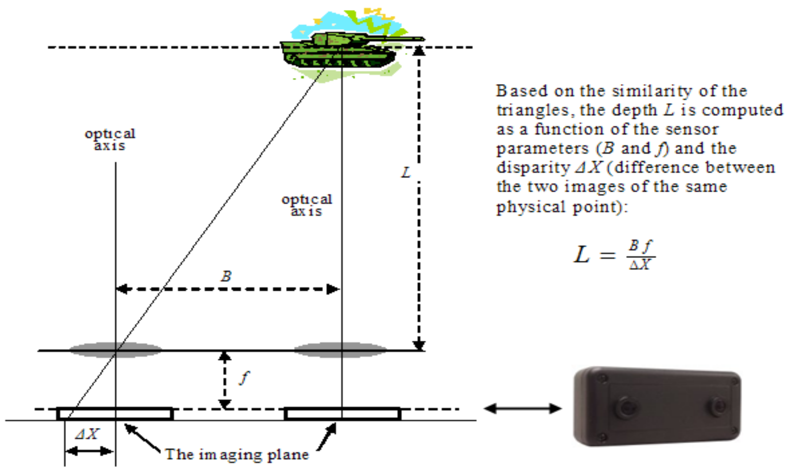

optical axis
optical axis
L
B
f
ΔX
The imaging plane
Based on the similarity of the triangles, the depth L is computed as a function of the sensor parameters (B and f) and the disparity ΔX (difference between the two images of the same physical point):
L = Bf/ΔX

Fig. 1. Stereoscopic vision system

In this paper, we propose a stereo imaging approach with uncalibrated camera(s), which is based on Gabor filter, improved Elastic Graph Matching, and multi-view geometry theories. The diagram of our proposed system is illustrated in Fig. 2. Compared with the current methods, the advantages of our approach include accurate feature detection and registration, stereo imaging without image deformation, and low computational complexity for potential real-time deployment.

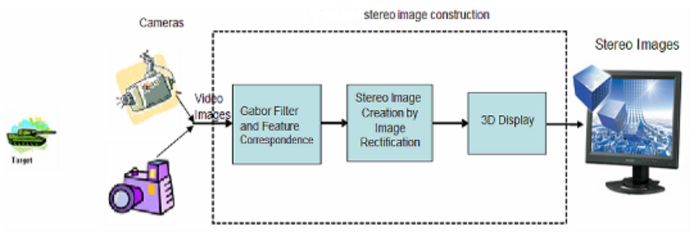

Cameras
stereo image construction
Stereo Images
Video Images
Gabor Filter and Feature Correspondence
Stereo Image Creation by Image Rectification
3D Display
Target

Fig. 2. Conceptual diagram of our proposed stereo imaging system

2 Algorithm Description

The proposed approach consists of three sequential steps. The first step is Gabor filter-based feature extraction which provides an efficient way for image feature extraction. The second step is an improved Elastic Graph Matching (EGM) to find feature correspondence of input image pair. The last step is stereo image generation based on the theories of multi-view geometry.

2.1 Gabor Filter

In human visual system (HSV), research has shown that people are sensitive to both specific orientation and spatial frequencies of the target of interest. For feature representation and extraction, wavelets are good at modeling both the orientation and frequency characteristics of object of interest. A Gabor filter bank can act as a simple form of wavelet filter bank. Because of its simplicity and optimum joint spatial/spatial-frequency localization, Gabor filter has attracted many research efforts [2], [3] and has been applied in many image analysis and computer vision-based applications, e.g. face recognition and fingerprint verification.

Gabor-filter bank is a group of 2-D filters which capture the optimal jointed localization properties of region of interest in both spatial and spectral domain. Typically, an image is filtered with a set of Gabor filters which have different or preferred orientations and spatial frequencies. To be specific, an image $I(\vec{x})$ is filtered with a set of Gabor wavelets as follows,

$$(wI)(\vec{k},\vec{x}_0) = \int \phi_{\vec{k}}(\vec{x}_0 - \vec{x}) I(\vec{x}) d\vec{x}$$

where $\phi_{\vec{k}}$ is the Gabor wavelet (filter) defined by

$$\phi_{\vec{k}}(\vec{x}) = \frac{\vec{k}}{\sigma^2} \exp(-\frac{\vec{k}^2 \vec{x}^2}{2\sigma^2})[\exp(i\vec{k}\vec{x}) - \exp(-\frac{\sigma^2}{2})]$$

with $\vec{k} = k_v e^{i\phi_\mu}$ controlling the orientation and the scale of the filters. By varying v and μ, we can get different Gabor filters with different orientations and scales. In our implementation, μ controls the orientation and is assigned by any value of 0, 1, 2, to 7 and v controls the spatial frequency and is assigned from 0, 1, and 2 with $k_v = (\pi / 2) / \sqrt{2^v}$ and $\phi_\mu = (\mu\pi)/8$. After filtering with a set of Gabor filters (24 filters from the above choice of v and μ), the outputs on each pixel in the image form a 24-dimensional vector called "jet". The amplitude of the jet represents whether a pixel has significant gradient value in both orientation and frequency. Thus, it can be used to determine if this pixel is a good feature for matching and tracking. In [6], some more complicated criteria are given for selecting good feature.

2.2 Feature Correspondence

Given the scenario that stereo images are generated from two videos (image sequences) acquired at two different views or one video (image sequence) taken by one single moving camera, we need to solve the feature correspondence problem to each synchronized image pair of two views or the image pair selected from single mobile camera, which can be thought as the most important step in stereo imaging with uncalibrated camera(s). With the Gabor-filter based features, we develop an improved version of the Elastic Graph Matching (EGM) method to identify the correspondence to the image pair. EGM [3], [4] has been applied successfully in many applications. But, due to the possible arbitrary relative position between the sensors, different camera settings, and different distances to the scene of interest, the conventional EGM methods may never converge to the correct position because of the position, orientation, and scale difference between the two images of the pair, and thus we propose to roughly match the image pair first, and then use EGM method to tune the matching result further. The matching between two images with unknown rotation and size can be formulated using a non-orthogonal image expansion approach [5]. One important issue in stereo matching is that the same object might have different 2D projections at different viewpoints. Here, we assume the difference of the projections is not big. The assumption is reasonable to most cases, which is based on the fact that a single moving camera has high frame rate (>15frames/sec) or the inspected scene is far away from the two different-view cameras. For small or middle-level shape changes, called shape deformation, our modified EGM can work efficiently because of the philosophy of EGM which is designed specially for the deformed-shape matching. Note that to guarantee the correctness of using this modified EGM, we require the image pair for stereo imaging must have at least ½ overlap and the EGM is only applied to overlap.

The main steps of feature correspondence are shown below:

Step 1: Find approximate position: We use the novel template matching with unknown rotation and size parameter [5] to identify the initial correspondence/matching between a target and the template of reference image/database. From the correspondences, some corresponding pairs of pixels from target and template are selected as features whose magnitudes of the jets are obviously larger than that of other pixels.

Step 2: Verify position: We first average the magnitudes of the jets of each feature point. The jets to each pixel are termed as "bunch". Then, we assign the average value to the processed bunch and compute the similarity function S_a without phase comparison.

$$S_a(J,J^{'}) = \frac{\sum_j a_j a_j^{'}}{\sqrt{\sum_j a_j^2 \sum_j a_j^{'2}}}$$

where a_j is the average value of the jth bunch. Alternatively, we can compute the similarity function S_ϕ with phase.

$$S_\phi(J,J^{'}) \approx \frac{\sum_j a_j a_j^{'} \cos(\phi_j - \phi_j^{'})^2}{\sqrt{\sum_j a_j^2 \sum_j a_j^{'2}}}$$

If the similarity is larger than a predefined threshold, the result by template matching is acceptable. Otherwise, error message will be generated and the EGM process is stopped.

Step 3: Refine position and size: To the current bunch graph, we vary its position and size to tune the correspondence. For each bunch, we check the four different pixels (± 3 , ± 3) displaced from its corresponded position in the target image. At each position, we check two different sizes with a factor of 1.2 smaller or larger the bunch graph.

Step 4: Refine aspect ratio: A similar relaxation process as described in Step 3 is performed. But at this time, we apply the operation only to x and y dimensions respectively.

2.3 Stereo Image Creation

It is not a trivial problem to create a pair of stereo images, even after finding the feature correspondence of the two images from uncalibrated cameras. With the knowledge of the correspondence of the two images, we need to rectify the two images of general viewpoints to form a stereo pair so that the stereo images look the same as if they were taken from a true stereo camera system. Conventional image rectification research focuses primarily only on making stereo matching easier but pays little attention to whether the rectified images form a natural stereo pair like those from a stereo camera (which is critical to the 3D display application). Here we present a rectification scheme that makes the rectified images look like a true stereo image that satisfies the constraints of the 3D display, and hence enables stereoscopic visualization of two general views of the same scene.

2.3.1 Standard Stereo System

We first consider the standard stereo setup, as shown in Fig. 3, which satisfies the following constraints:

1) The two optical axes are parallel and perpendicular to the baseline.
2) The two cameras have the same intrinsic parameters.
3) Two image planes are aligned, which means that the x axes of the two images are parallel to the baseline.

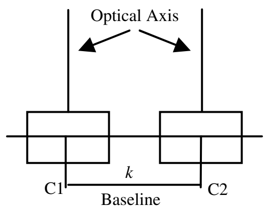
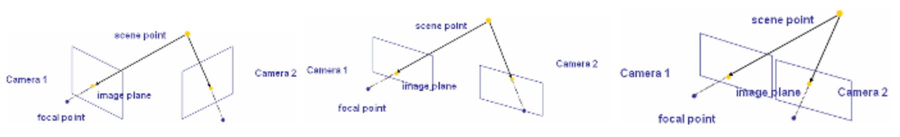


Fig. 3. A standard stereo setup

The stereo imaging for standard stereo system can be found in many references and text books.

2.3.2 Solution to Uncalibrated Camera

Here we propose an efficient algorithm for constructing a stereo image from two different-view images. For any two video frames or two different-view images, the corresponding camera(s) setup may vary or are different during data acquisition and the constraints listed above may not be satisfied anymore. Therefore, our task of stereo imaging can be defined as rectifying the images so that they appear to come from a standard stereo camera. It can be easily shown that, for pure rotation or any internal parameter change, there exists a homography transformation for image rectification, which means that we do not have to translate the cameras directly to set a stereo camera (it is not feasible in real life), but we can achieve this by rectifying the images with a homography transformation. That is to say, we can "rotate" and "change camera matrix" by applying a proper homography to the two images, so that the two images are transformed to a stereo pair identical to one captured by a standard stereo camera. In summary, our problem of constructing a stereo image from a single moving camera can be defined as: Given two video frames captured at two general viewpoints, called "Camera 1" and "Camera 2" as illustrated in Fig. 4 (a), we aim at getting a true stereo pair from these two images through image rectification. The rectification process can be broken down into two steps. First, by finding and applying a homography to each image, we transform these two images to the new ones, which are identical to the ones captured by two parallel cameras, as illustrated in Fig. 4 (b). Second, we adjust the wide or narrow baseline of the two parallel cameras to a proper value (say standard base line) by translating the new images with a proper value. Thus, the desired stereo pair is constructed in Fig. 4 (c).



(a) Two general image plane (b) Rectified image planes (c) Stereo image pair with proper baseline

Fig. 4. Image Rectification

- **Transforming Each Image By Its Homography**

In this case, the epipolar constraint [7] is followed. That is, after rectification, we should have the following properties: (1) Any epipolar line in each image should be horizontal. This means that the epipole is at $(k,0,0)^T$. (2)The corresponding epipolar lines should be the same. This means that, given two corresponding epipolar lines $\{l,l'\}$ such that $l' = F[k]_\times l$, l' should equal to l. There are already some methods [7-11] for stereo imaging based on multi-view geometry. Here, we give a very concise description for stereo imaging which is the summary of the above methods. To make the stereo images look like standard stereo setup as much as possible, we give the following major steps for stereo image generation:

Algorithm. Stereo Image Generation from Uncalibrated Camera

Step 1. Estimate fundamental matrix *F* which represents the difference between the two images

Step 2. Compute the rectification matrix H_2 for the second image

Step 3. Compute the rectification matrix H_1 for the first image

Step 4. Rectify each pixel of the two selected images by multiplying H_1 and H_2 respectively to generate wide-baseline stereo pair

Step 5. Compute the average Z value of the center part of image and translate image to configure a proper baseline for 3D display

- **Improving Stereo Pair Quality By Accurately Computing {a, b, c}**

In the stereo imaging via multi-view geometry, matrix *F* and H_2 can be calculated accurately, according to [7-11]. But, for H_1, we need to estimate the matrix H_A, which is the unknown component of H_1 in the form of $(a,b,c;0,1,0;0,01)$. In conventional methods, such as [7-11], solving for H_A is not clearly addressed. Sometime, a solution would be given, but is only suitable to stereo matching, and therefore not applicable to our case. We have to develop some criteria to determine the vector (a, b, c) more accurately while still being suitable to our case. Conventional methods use a criterion that minimizes the disparity of the selected point pairs for stereo matching. This criterion cannot be used for our case as we want the results undistorted with respect to a standard stereo pair. Therefore, we propose to use a new minimization criterion to estimate vector (a, b, c). First, we first segment the image pair into many regions according to the image homogeneity. Second, we select some feature points in the first image, such as edge points, from each region and find their correspondence in the second image. Third, because the feature points with same depth in the same region have same disparity, we formulize a minimization equation to compute (a, b, c).

$$mim(\sum_{p} \sum_{i,j\in A_p} (\left\|H_1 x_i - H_1 x_j\right\| - \left\|H_2 x_i' - H_2 x_j'\right\|)^2)$$

Here A_p is a set of points such that for $i, j \in A_p$, x_i and x_j have the same depth and in the same image. x_i and x_i' is the corresponding pair in the two image.

3 Experimental Results

Extensive tests including indoor/outdoor scenarios were performed to validate the efficiency of our proposed algorithm. 3D video generation for moving target inspection was also performed. **Note that all the stereo results illustrated in the paper are in red-cyan format. Reader can perceive these 3D images with regular red-cyan 3D glasses.**

3.1 Tests with a Single Moving Camera

A set of multiple-view images was taken by a hand-hold moving camera. The resolution of the true-color image is 1024x768 and the internal parameters (such as focal length, aspect ratio, lens, shutter speed, etc.) of camera are unknown. The relative position and the orientation of the multi-views were not calculated.

- **Test in Indoor Environment**

Indoor tests were performed with a mobile camera. Fig. 5 shows the process and results. It can be seen (with 3D glasses) that the stereo image has been successfully created.

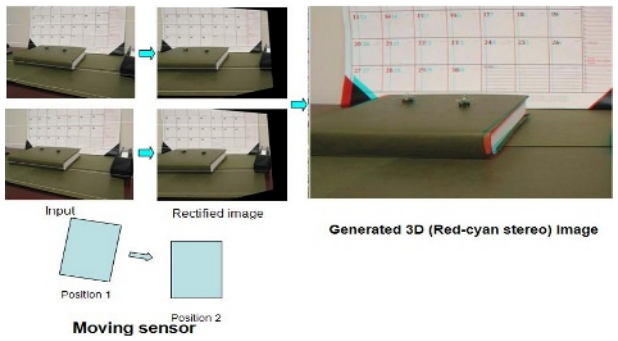


Fig. 5. Example of indoor test

- **Test in Outdoor Environment**

Outdoor images were taken with objects hiding in grasses. Fig. 6 shows that a 3D image was successfully created. 3D effect of the scene can be easily perceived with 3D glasses.

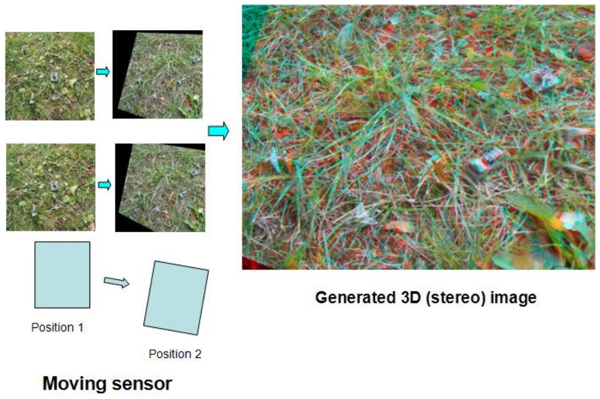


Fig. 6. Test in out-door environment

- **Test on Depth Perception**

Two images at two different viewpoints were captured and then a stereo image was formed. Fig. 7 illustrates the result.

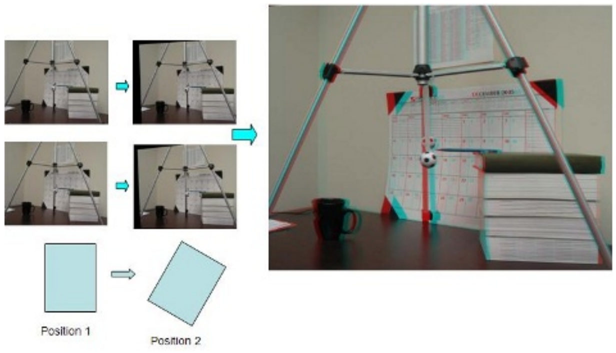


Fig. 7. Test on 3D depth perception

3.2 Moving Target Inspection with Two Video Cameras

Several video clips were captured by two ordinary video camcorders set around the scene of interest. Each video clip is at least 15 seconds long. The resolution of the color video is set as 640x480 and the internal parameters (such as focal length, aspect ratio, lens, shutter speed, etc.) of camcorder are unknown. The relative position and orientation of two cameras are also uncalculated. In the test, a moving target (pen) was moving forward and backward between two hanging balls. The Z-direction movement can not be observed in 2D video. But, with 3D video, the Z-direction movement can be easily perceived. Fig. 8 and Fig. 9 illustrate the 3D results.

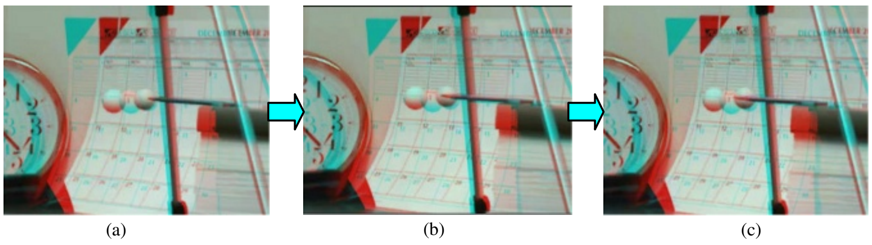


Fig. 8. Pen moving toward front: (a), (b), and (c) are red-cyan stereo images for different time instances

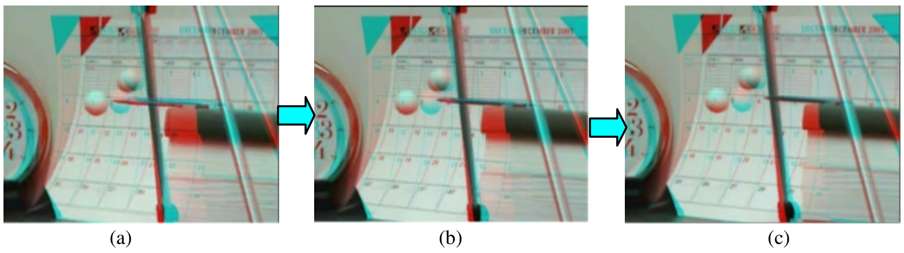


Fig. 9. Pen moving toward back: (a), (b), and (c) are red-cyan stereo images for different time instances

4 Conclusions

We have presented a systematic approach for stereo image generation. The approach consists of robust feature generation via Gabor filters, accurate feature correspondence by EGM, and reliable stereo image creation. Tests with real-life data clearly demonstrated the efficacy of our proposed method.

Acknowledgements

This work was supported by a grant from the U. S. Missile Defense Agency. The authors would like to thank Mr. Richard Felner and Mr. Steve Waugh for their helpful suggestions and comments.

References

1. Andrea Fusiello, Emanuele Trucco, Alessandro Verri: A compact algorithm for rectification of stereo pairs. Journal of Machine Vision and Applications, Vol. 12, (2000) 16-22
2. D. A. Clausi and H. Deng: Fusion of Gabor filter and co-occurrence probability features for texture recognition. IEEE Trans. of Image Processing, Vol. 14, No. 7, (2005) 925-936
3. L. Wiskott, J. Fellous, N. Kruger, C. Malsburg: Face recognition by Elastic Bunch Graph Matching, Intelligent Biometric Techniques in Fingerprint and Face Recognition, CRC Press, Chapter 11, (1999) 355-396
4. D. Beymer: Face recognition under varying pose. In Proc. IEEE Conf. Computer Vision and Pattern Recognition. (1994) 756-761
5. R. M. Dufour, E. L. Miller, N. P. Galatsanos: Template matching based object recognition with unknown geometric parameters. IEEE Trans. on Image Processing, Vol.11, (2002)
6. J. Shi and C. Tomasi: Good Features to Track. Proc. IEEE Conference on Computer Vision and Pattern Recognition. (1994)
7. R. Hartley and A. Zisserman: Multiple view geometry in computer vision. Cambridge University, Cambridge, 2nd edition, (2003)
8. R. Hartley: Theory and practice of projective rectification. IJCV, 35(2) (1999) 1-16
9. C. Loop and Z. Zhang: Computing rectifying homographies for stereo vision. In CVPR, (1999) 125-131
10. P. May: A Survey of 3-D Display Technologies. Information Display, Vol.32 (2005)
11. Z. Zhang: Determining the epipolar geometry and its uncertainty: a review. In IJCV, 27(2): (1998) 161–195

Global Hand Pose Estimation by Multiple Camera Ellipse Tracking

Jorge Usabiaga[1], Ali Erol[1], George Bebis[1],
Richard Boyle[2], and Xander Twombly[2]

[1] Computer Vision Laboratory, University of Nevada, Reno, NV 89557
{usabiaga, aerol, bebis}@cse.unr.edu
[2] BioVis Laboratory, NASA Ames Research Center, Moffett Field, CA 94035
Richard.Boyle@nasa.gov, xtwombly@mail.arc.nasa.gov

Abstract. Immersive virtual environments with life-like interaction capabilities have very demanding requirements including high precision and processing speed. These issues raise many challenges for computer vision-based motion estimation algorithms. In this study, we consider the problem of hand tracking using multiple cameras and estimating its 3D global pose (i.e., position and orientation of the palm). Our interest is in developing an accurate and robust algorithm to be employed in an immersive virtual training environment, called "Virtual GloveboX" (VGX) [1], which is currently under development at NASA Ames. In this context, we present a marker-based, hand tracking and 3D global pose estimation algorithm that operates in a controlled, multi-camera, environment built to track the user's hand inside VGX. The key idea of the proposed algorithm is tracking the 3D position and orientation of an elliptical marker placed on the dorsal part of the hand using model-based tracking approaches and active camera selection. It should be noted that, the use of markers is well justified in the context of our application since VGX naturally allows for the use of gloves without disrupting the fidelity of the interaction. Our experimental results and comparisons illustrate that the proposed approach is more accurate and robust than related approaches. A byproduct of our multi-camera ellipse tracking algorithm is that, with only minor modifications, the same algorithm can be used to automatically re-calibrate (i.e., fine-tune) the extrinsic parameters of a multi-camera system leading to more accurate pose estimates.

1 Introduction

Virtual environments (VEs) should provide effective human computer interaction (HCI) for deployment in applications involving complex interaction tasks. In these applications, users should be supplied with sophisticated interfaces allowing them to navigate in the VE, select objects, and manipulate them. Implementing such interfaces raises challenging research issues including the issue of providing effective input/output. At the input level, new modalities are necessary to allow natural interaction based on direct sensing of the hands, eye-gaze, head or even the whole body.

Computer vision (CV) has a distinctive role as a direct sensing method because of its non-intrusive, non-contact nature; on the other hand, it is also facing various challenges in terms of precision, robustness and processing speed requirements. Various solutions have been proposed to support simple applications (i.e., no intricate object manipulation) based on gesture classification and rough estimates of almost rigid hand motion. However, systems that can support advanced VE applications with life-like interaction requirements have yet to come. Applications such as immersive training or surgical simulations require very accurate and high frequency estimates of the 3D pose of the hand in a view independent fashion (i.e., the user need not even know where the cameras are located). Recovering the full degrees of freedom (DOF) hand motion from images with unavoidable self-occlusions is a very challenging and computationally intensive problem [2][3].

This study is part of an effort to improve the fidelity of interaction in an immersive virtual environment, called "Virtual GloveboX" (VGX) [1], which is currently under development at NASA Ames (see Fig. 1). Our objective is to employ computer vision-based hand motion capture. VGX is being designed to assist in training astronauts to conduct technically challenging life-science experiments in a glovebox aboard the International Space Station. It integrates high-fidelity graphics, force-feedback devices, and real-time computer simulation engines to achieve an immersive training environment.

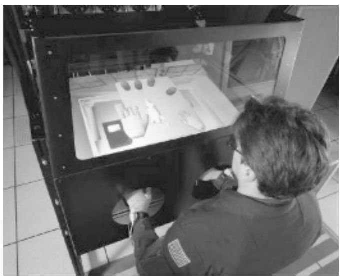


Fig. 1. Virtual Glove Box: A stereoscopic display station provides a high-resolution immersive environment corresponding to a glovebox facility. The users interact with virtual objects using datagloves.

The effectiveness of VGX as a training tool depends both on precision of the sensed motion and ease of use. The current interface of VGX uses off-the-shelf tracking and haptic feedback devices which contain cumbersome elements such as wired gloves, tethered magnetic trackers, and haptic armatures inside the workspace. All of these hinder the ease and naturalness with which the user can interact with the computer controlled environment and calibration of each measured degree of freedom is time consuming and imprecise. Further research is thus required to reduce the need for encumbered interface devices and increase the value of VGX as a training tool.

A fully generic unconstrained and precise solution to the hand pose estimation is not available yet. Existing unadorned hand tracking systems are mostly limited to a single camera and implicitly or explicitly accompanied with a number of viewing constraints to minimize self-occlusions [2][3]. Obviously, such approaches are not acceptable in this and similar applications. Although some marker-based approaches are available, precision issues are often not addressed in these studies.

In this paper, we present an marker-based 3D global hand pose (i.e., position and orientation of the palm) estimation system that operates in a multi-camera environment built to track the user's hand inside the VGX. The use of markers is well justified in the context of our application since VGX naturally allows for the use of gloves without disrupting the fidelity of the interaction. Moreover, users are not looking at their hands during the simulation but at graphical hand models displayed in the virtual environment (see Fig. 1).

Estimating the global pose of the hand has several advantages. First, it reduces the dimensionality of hand pose estimation by 6 DOF. Second, it is a requirement for inverse kinematics-based methods. Finally, for some interfaces (e.g. navigation in VE), estimating the rigid motion of the hand is sufficient to generate control signals for the application. Our experimental results illustrate that the proposed approach is more accurate and robust than related approaches. A byproduct of our multi-camera ellipse tracking algorithm is that, with only minor modifications, the same algorithm can be used to automatically re-calibrate (i.e., fine-tune) the extrinsic parameters of a multi-camera system. In our case, camera re-calibration leads to improved hand pose estimates.

The rest of the paper is organized as follows: in the next Section, we present a brief review of previous work on marker-based hand pose estimation approaches. In Section 3, we describe the multiple camera environment used track the hand in the context of our application. In Sections 4 and 5, we provide detailed descriptions of the multiple camera ellipse tracking algorithm and its application to camera re-calibration. Section 6 presents our experimental results and comparisons. Finally, Section 7 concludes this study.

2 Previous Work

Marker-based hand tracking is not a very common approach due its intrusive nature. Nevertheless, there have been many attempts using point markers [4,5,6,7,8,9]. Placing a number of markers on the dorsal surface of the hand, fingertips and/or joints can provide valuable information that can be used to estimate joint angles by solving an inverse kinematics problem. In [6], Holden applied model-based tracking using fingertip and joint markers for ASL recognition. Lien et al. [7] and Lee [8] used stereo cameras to extract the 3D locations of a number of markers on the palm and fingertips and then applied Genetic Algorithms (GAs) to estimate the orientation of the palm. The state of the fingers was estimated using inverse kinematics and regression techniques. In [4], closed form solutions were derived to calculate the angles from 2D marker positions under orthographic projection. In a more recent study, Kim et al. [9] used white fingertip markers under black-light and stereo cameras to extract 3D fingertip locations for gesture classification.

The main problem with point markers is their susceptibility to occlusions and localization difficulties. Because of the proximity and flexibility of fingers, loosing some of the markers completely or collision of the markers on the image plane are very likely events that increase the complexity of tracking [6]. Moreover, when the hand is allowed to move in a relatively large area, it is not feasible to use point markers due to localization errors which affect the precision of pose estimates. In the case of fingers, it is not quite possible to use other than point or line markers, which do not guarantee good precision and robustness due to practical resolution constraints and abundance of these features in images. The palm, however, is large enough allowing the use of more robust markers. Among them, conics have often proved to be good candidates due to several following reasons [10]. First, like points or straight lines, they are preserved under perspective and projective transformations. Second, conics are more compact primitives which contain global information of an object's pose. Finally, a conic can be represented by a symmetric matrix which is easy to manipulate. In some cases, a closed-form solution [10,11] can be obtained, avoiding more expensive non-linear iterative techniques.

To the best of our knowledge, Maggioni et al. [12] is the only study using conics, (i.e., two concentric circular markers) for estimating global hand pose in 3D. Viewing the markers from a single camera is sufficient to obtain the orientation and position of the palm.

3 Operational Environment

The glovebox environment has some features that can be easily exploited by vision-based algorithms for hand tracking. First, the users are expected to wear gloves, which enables the use of markers naturally. Second, hand motion is restricted to a relatively small area inside the glovebox. This justifies the use of multiple cameras to deal with occlusions and controlled lighting along with uniform background to enable segmentation of the hands. Taking these facts into consideration, we have built a mock-up of VGX to perform our experiments as shown in Figure 3.

Specifically, the VGX mock-up contains 8 hardware-synchronized cameras located at the corners of the box, several fluorescent lights, and a white background to help segmenting the hands. The intrinsic parameters of the cameras and radial distortion parameters were calibrated using Matlab's Calibration Toolbox [13]. To estimate the extrinsic camera parameters, Svoboda's [14] multiple camera self-calibration procedure was used.

During simulation, users wear a glove with an elliptical marker placed on the dorsal part of the palm. In principle, it is possible to estimate the pose of the hand using two coplanar ellipses [10], however, resolution limitations combined with un-constrained hand motion can make it difficult to locate each ellipse separately. Therefore, we decided to use a single ellipse, which would need to be visible from at least two cameras for estimating its pose [11]. Camera placement in the VGX mock-up satisfies this visibility constraint.

4 Multiple-Camera Ellipse Tracking

Ellipse pose estimation is a well studied topic and there exist several efficient algorithms for estimating pose information using one or two cameras under certain conditions. In our initial experiments, we found Quan's algorithm [11] using two views of a single ellipse to be very efficient, fast, and accurate. This algorithm deals with the problem of conic correspondences and reconstruction in 3D from two views using projective properties of quadric surfaces. A closed-form solution for both projective and Euclidean reconstruction of conics as well as a mechanism to select the correct two ellipses in each of the two images are described in [11].

The use of multiple cameras was deemed necessary in our application to allow hand tracking independent of viewpoint. In our system, the elliptical marker could be visible from up to four cameras. Although not all of the cameras would contain reliable information for pose estimation (i.e., see Section 4.1), it would be possible in general to use information from more than two cameras to improve pose estimation and robustness. Therefore, we have developed a model-based hand tracking approach that integrates information from any number of cameras.

In model-based tracking, at each frame of an image sequence, a search in the parameter space is performed to find the best parameters that minimize a matching error between groups of model features and groups of features extracted from the input images. In the case of multiple cameras, the errors over all the available views are accumulated. The search is often initiated by a prediction based on the dynamics of the system. In the first frame, however, a prediction is not available and a separate initialization procedure is required.

In our system, we have used Quan's algorithm [11] for initialization purposes. There are many different ways to conducting the search or equivalently minimize the matching error. Here, we present an algorithm based on Martin and Horaud's [15] extension of Lowe's model-based pose estimation algorithm [16]. Specifically, there are three main processing steps in our algorithm: (1) active camera selection, where the best cameras for pose estimation are determined, (2) matching error computation, where the similarity between the projected model ellipse and the image features is calculated, and (3) pose estimation, where the matching error is minimized.

4.1 Active Camera Selection

We use a number of criteria to select the "best" cameras for pose estimation. First, we select only those cameras that provide us with images of the ellipse at a satisfactory resolution. If the ellipse is too far away, large changes in its pose will only cause small image displacements. The criterion used to test this constraint is the area covered by the ellipse in the image. Second, we try to avoid selecting cameras that provide an image where the contour of the ellipse is too close to the silhouette of the hand. The criterion used for this is the angle between the normal to the ellipse and the vector that goes from the center of the camera to the center of the ellipse. Finally, we do not consider cameras that provide images where the ellipse is completely or partially occluded.

4.2 Computation of Matching Error

The ellipse model is represented by a set of uniformly sampled points on its boundary. For each active camera i, a signed error vector e^i is computed by (1) projecting the m points onto the camera's image plane using the current prediction of the pose of the ellipse, and (2) searching for the maximum gradient along the normal to the projected contour at the sampled points. The errors of all the points are concatenated to form a vector:

$$\boldsymbol{e^i} = \left[\boldsymbol{e^i_1}, ..., \boldsymbol{e^i_m}\right]^{\boldsymbol{T}} \tag{1}$$

where i denotes camera i. It should be noted that, a large number of sample points m would provide a better estimate; however, it would also slow down the system significantly.

4.3 Pose Estimation

Pose estimation corresponds to finding the pose parameters T (i.e., position and orientation of the ellipse) that minimize the matching error. Many studies [16] [15] employ Newton's method which subtracts a vector of corrections, x from the current estimate for T at each iteration. If T^k is the parameter vector corresponding to iteration k, then,

$$\boldsymbol{T^{k+1}} = \boldsymbol{T^k} - \boldsymbol{x} \tag{2}$$

By linearizing the system at the current estimate, the correction vector is calculated by solving an over-determined system of equations:

$$\boldsymbol{e} = \boldsymbol{Jx} \tag{3}$$

where J is the Jacobian. The total error vector e is is obtained by weighting and concatenating the error vectors (see Eq. 1) of the active cameras given by:

$$\boldsymbol{e} = \left[\boldsymbol{w^1 e^1}, ..., \boldsymbol{w^n e^n}\right]^{\boldsymbol{T}} \tag{4}$$

where the weights w^i are calculated as a combination of (1) calibration error (i.e., the larger the calibration error the smaller the weight), and (2) area (i.e., the larger the area covered by the ellipse on this camera's image the larger the weight). Weighting mainly helps to reduce the number of iterations required by the algorithm to converge an it does not really improve results.

5 Extrinsic Parameters Re-calibration

Multiple camera calibration assuming an arbitrary camera configuration is a difficult problem. Svoboda's [14] approach provides a relatively practical solution. Instead of a complex calibration pattern, it uses a colored light source (e.g., a small LED in our case), which is visible by many cameras simultaneously. Calibration is performed by moving the light source arbitrarily inside the area covered by the cameras. The trajectory of the light source as perceived from different cameras provides the necessary information for calibration purposes. However, this process is rather slow, it requires some user interaction, and it does not always guarantee good results since it depends on how well the trajectory of the light source covers the area enclosed by the cameras.

Re-calibration of a multi-camera system could be necessary for many reasons, for example, when the cameras move. In this case, even a slight variation in the position or orientation of the cameras could affect pose estimation. To update and further optimize the extrinsic camera parameters, we have employed our ellipse tracking algorithm. Specifically, re-calibration works as follows:

1. **For all frames** run the tracking algorithm and record (i) the pose of the ellipse and (ii) which cameras are active for each frame (see Table 1, top).
2. **For each camera**, load the poses, images, and frames p where this camera was active; we will be referring to these frames as **active** frames (see Table 1 bottom).
3. Using this information, compute the errors as explained in 4.2, however, instead of putting them in a vector, add their absolute values (m is the number of samples per frame):

$$\boldsymbol{e^i} = \sum_{\boldsymbol{k=1}}^{\boldsymbol{p}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{m}} \mid \boldsymbol{e^i_{kj}(x^i)} \mid \tag{5}$$

   where k indicates the frame number, j indicates the sample number, and i indicates the camera number. e^i_{kj} is a function of the extrinsic camera parameters x^i. The Nelder-Mead's Simplex algorithm [17] was used to find the Δx^i that minimizes the error e^i.
4. Go to step 1 until the error is smaller than a threshold or a maximum number of iterations has been reached.

6 Experimental Results

In this section, we present quantitative and visual experimental results to evaluate the pose estimation and re-calibration algorithms. In all the experiments, we assumed that the ellipse was placed flat on the dorsal part of the hand (see Fig. 2).

Table 1. Example for a sequence with 999 frames. **Top** Recover the ellipse pose for all frames. **Bottom** Run re-calibration for all cameras (A = Active camera/frame; I = Inactive camera/frame).

1- Run ellipse tracking for whole training sequence

Frame	Camera					Ellipse pose	
$\Downarrow$	0	1	...	7		Orientation	Position
1	I	I	...	A	$\rightarrow$	θ_1	p_1
2	I	I	...	A	$\rightarrow$	θ_2	p_2
⋮		⋮			$\rightarrow$	⋮	⋮
997	A	A	...	I	$\rightarrow$	θ_{997}	p_{997}
998	A	A	...	I	$\rightarrow$	θ_{998}	p_{998}
999	A	A	...	I	$\rightarrow$	θ_{999}	p_{999}
Total	300	351	...	415			

2- Run re-calibration for all cameras

Camera	Frame					Extrinsic Parameters	
$\Downarrow$	0	1	...	999		Rotation	Translation
0	I	I		A	$\rightarrow$	R_0	t_0
1	I	I		A	$\rightarrow$	R_1	t_1
2	I	I		I	$\rightarrow$	R_2	t_2
3	I	I	...	I	$\rightarrow$	R_3	t_3
4	A	A		I	$\rightarrow$	R_4	t_4
5	I	I		A	$\rightarrow$	R_5	t_5
6	I	I		A	$\rightarrow$	R_6	t_6
7	A	A		I	$\rightarrow$	R_7	t_7

6.1 Accuracy of Ellipse Pose Estimation

To evaluate the accuracy of pose estimation, we compared our algorithm with Quan's ellipse pose estimation algorithm, which, in our opinion, is the best available algorithm for a two camera system. To be able to use Quan's algorithm in our multi-camera environment, we used the same criteria given in section 4.1 for selecting the best two cameras.

Fig. 2 shows the re-projection error (i.e., matching error given in section 4.2) for both algorithms. The square wave shaped curve on the top of the graph indicates the number of active cameras at each frame. Two interesting observations can be made:

1. When only two cameras are active in the case of the multiple-camera algorithm, both algorithms give very close results. However, when more than two cameras are active, the performance of the multiple-camera algorithm is significantly better.

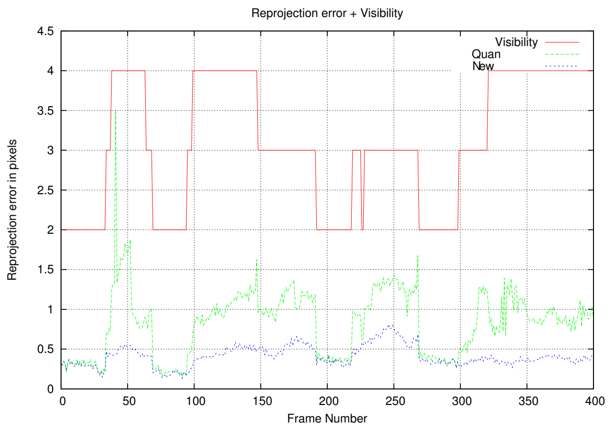


Fig. 2. The re-projection error of Quan's algorithm and our algorithm

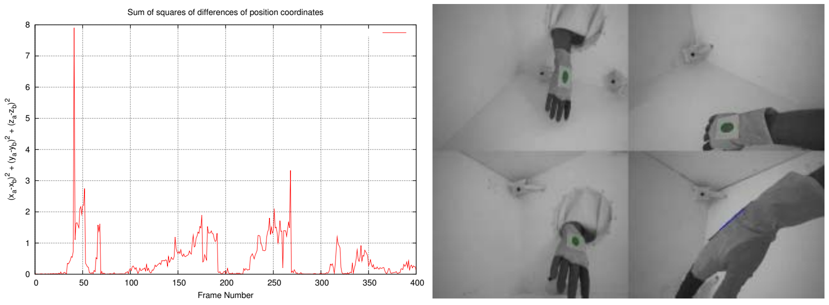


Fig. 3. (Left) Sum of squares differences of position coordinates between two and multiple-camera algorithms. (Right) Re-projection of the ellipse on the input images for frame 42 of a sequence. The images where the re-projected ellipse is drawn in green correspond to the active cameras.

2. Although the re-projection error is smaller in the multiple-camera case, it increases with the number of cameras. The reason is that there are more calibration errors involved as the number of cameras increases.

Fig. 3(Left) shows the differences in the position estimates of the two algorithms. Interestingly enough, these differences resemble the differences in the re-projection error shown in Fig. 2. Overall, we can conclude that when both algorithms use the same two cameras, the results are very similar, however, when more cameras are available, the multiple-camera approach yields more accurate position estimates which implies lower re-projection error. Similar observations can be made for the orientation estimates of the ellipse.

Finally, Fig. 3(Right) shows several examples to demonstrate our multiple-camera algorithm in the case of three active cameras.

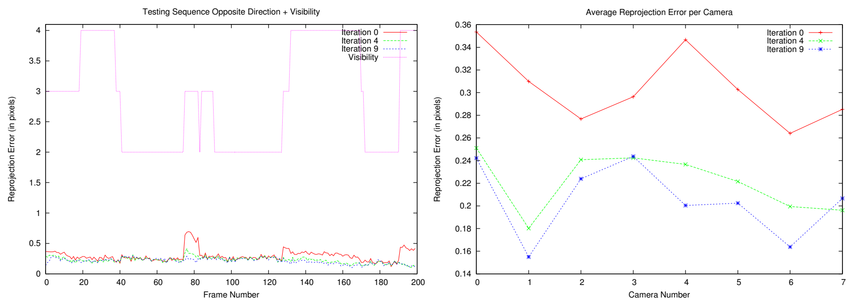


Fig. 4. The re-projection error on a testing sequence with and without re-calibration. (Left) Average re-projection error per frame for the testing sequence computed with the extrinsic parameters iterations 0, 4 and 9. (Right) Average error of the whole sequence per camera.

6.2 Processing Speed

A disadvantage of the multiple camera tracking system is the higher computational requirements due to its iterative nature. Quan's algorithm processes each frame in about 4 ms. The multiple camera algorithm deals with more cameras and computational cost depends linearly on the number of sampled points of the ellipse used for re-projection. Using a rather conservative number of samples (100) and un-optimized code, the processing speed was about 150 ms per frame. The most expensive part of the algorithm is the matching error calculation step, which is repeated a few times at each frame.

6.3 Effects of Re-calibration

The results of the multiple-camera algorithm on a sequence taken in the VGX were used to re-calibrate the extrinsic camera parameters. To asses the effects of re-calibration, the modified extrinsic parameters were used to estimate the pose parameters on a different test sequence. Fig. 4 shows the re-projection errors obtained with and without re-calibration assuming different number of iterations. As it can be observed, lack of re-calibration increases the re-projection error as number of active cameras increases. However, re-calibration reduces the dependency of the re-projection error on the number of active cameras, to the point where it is almost constant.

7 Conclusion and Further Work

We have presented a multiple camera, model-based ellipse tracking algorithm for global hand-pose estimation in an immersive training environment. We have also shown how to employ the proposed algorithm for re-calibrating the extrinsic parameters of a multi-camera system. Our experimental results illustrate the effectiveness of the proposed approach both in terms of pose estimation and re-calibration. For future work, we plan to consider the problem of estimating the full DOF of the hand. Estimating the global pose of the hand is an important step in this process.

Acknowledgments

This work was supported by NASA under grant # NCC5-583.

References

1. Twombly, X., Smith, J., Montgomery, K., Boyle, R.: The virtual glovebox (vgx): a semi-immersive virtual environment for training astronauts in life science experiments. Journal of Systemics and Informatics **6** (2006)
2. Pavlovic, V.I., Sharma, R.S., Huang, T.S.: Visual interpretation of hand gestures for human-computer interaction: A review. PAMI (1997)
3. Erol, A., Bebis, G., Nicolescu, M., Boyle, R., Twombly, X.: A review on vision based full dof hand motion estimation. IEEE Workshop on Vision for Human Computer Interaction (2005)
4. Chua, C.S., Guan, H.Y., Ho, Y.K.: Model-based finger posture estimation. In: ACCV2000. (2000)
5. Chua, C., Guan, H., Ho, Y.: Model-based 3d hand posture estimation from a single 2d image. Image and Vision Computing **20** (2002) 191–202
6. Holden, E.: Visual Recognition of Hand Motion. PhD thesis, Department of Computer Science, University of Western Australia (1997)
7. Lien, C.C., Huang, C.L.: Model based articulated hand motion tracking for gesture recognition. Image and Vision Computing **16** (1998) 121–134
8. Lee, J., Kunii, T.: Constraint-based hand animation. In: Models and Techniques in Computer Animation. Springer, Tokyo (1993) 110–127
9. Kim, H., Fellner, D.W.: Interaction with hand gesture for a back-projection wall. In: CGI '04: Proceedings of the Computer Graphics International (CGI'04), Washington, DC, USA, IEEE Computer Society (2004) 395–402
10. Ma, S.: Conics-based stereo, motion estimation and pose determination. IJCV **10** (1993) 7–25
11. Quan, L.: Conic reconstruction and correspondence from two views. IEEE Transactions on Pattern Analysis and Machine Intelligence **18** (1996) 151–160
12. Maggioni, C., B., K.: Gesturecomputer - history, design and applications. In Cipolla, R., Pentland, A., eds.: Computer Vision for Human-Machine Interaction. Cambridge (1998) pp. 312–325
13. Bouguet, J.Y.: (Camera calibration toolbox for matlab)
14. Svoboda, T., Martinec, D., Pajdla, T.: (A convenient multi-camera self-calibration for virtual environments)
15. Martin, F., Horaud, R.: Multiple-camera tracking of rigid objects. In: INRIA. (2001)
16. Lowe, D.: Fitting parameterized three-dimensional models to images. PAMI **13** (1991) 441–450
17. (Gnu scientific library) http://www.gnu.org/software/gsl/.

Vision-Based Self-localization of Autonomous Guided Vehicle Using Landmarks of Colored Pentagons

Y.S. Kim[1], J.C. Kim[2], E.J. Park[3], and Joonwhoan Lee[3]

[1] Research & Development Division for Hyudai Motor Company & Kia Motors Corporation
dudtkal@yahoo.co.kr
[2] Department of Electronic Engineering, Seonam University, Namwon, Chonbuk 590-711, Korea
jckim@seonam.ac.kr
[3] Department of Electronic Engineering Chonbuk National University, Jeonju, Chonbuk 560-756, Korea
{for0511, chlee}@ chonbuk.ac.kr

Abstract. This paper describes an idea for determining self-organization using visual land marks. The critical geometric dimensions of a pentagon are used here to locate the relative position of the mobile robot with respect to the pattern. This method has the advantages of simplicity and flexibility. This pentagon is also provided with a unique identification, using invariant features and colors that enable the system to find the absolute location of the patterns. This algorithm determines both the correspondence between observed landmarks and a stored sequence, computes the absolute location of the observer using those correspondences, and calculates relative position from a pentagon using its five vertices. The algorithm has been implemented and tested. In several trials it computes location accurate to within 5.4 centimeters in less than 0.3 second.

Keywords: pentagon, camera calibration, self-localization, AGV(Autonomous Guided Vehicle).

1 Introduction

Self-localization is important for AGV. There have been diverse researches focused on the problem of indoor environment [1, 2]. The self-localization should not accumulate position errors, so that ultrasonic sensor and vision systems are preferred to dead-reckoning systems. Because CCD camera becomes cheaper and the vision system does not depend upon the surface characteristics of irregular reflectance differently from ultra sonic sensors, it will become a popular modality. But the major difficulty of vision sensor is originated from ambiguity due to illumination changes and geometric distortions. To overcome such inherent difficulties in the vision system, one can rely on sophisticated image analysis algorithms. Such approach, however, inevitably increases complexity and makes the system slow. Rather landmark-based approach is simple and robust even though its application is limited to well-defined indoor environments. In this article, we propose colored pentagons as landmarks. There can be many types of landmarks such as a circle with vertical bars, and colored rectangles [1]. These methods, however, may provide some constraints to the camera attached to an AGV. For example, the optical axis should pass the center of a circle in the former approach, and the height of camera should be aligned with that of landmark in the latter one. In such sense, proposed scheme is more flexible, because the only constraint is the landmark should be located in the field of view of a camera.

2 Proposed Landmarks

A pentagon on a plane has invariant features to perspective transforms as shown in Fig. 1 [3]. Equation (1) represents invariant features defined by perspective transforms.

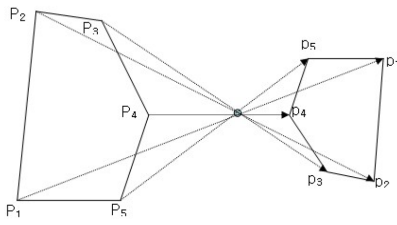


Fig. 1. Perspective Transform of a Pentagon

$$\left(\frac{|M_{431}||M_{531}|}{|M_{421}||M_{531}|}, \frac{|M_{421}||M_{532}|}{|M_{432}||M_{521}|}\right) = \left(\frac{|m_{431}||m_{521}|}{|m_{421}||m_{531}|}, \frac{|m_{421}||m_{532}|}{|m_{432}||m_{521}|}\right) \quad (1)$$

where matricies $M_{ijk}=(P_i, P_j, P_k)$ and $m_{ijk}=(p_i, p_j, p_k)$, p_i and p_i are the coordinates of a pair of corresponding vertices, and $|M_{ijk}|$ is the determinant of M_{ijk}. In order to simplify image processing and recognition steps, we impose following conditions for the shape of pentagons.

1) Distances of invariant features among pentagons are as large as possible.
2) Each pentagon has the same area.
3) Inner or outer angle at a vertex should not be close to 0 or 180 degrees.

Upon the above conditions, we recommend 5 pentagons as shown in Fig. 2. Note that rotated or mirrored versions of the pentagons can also be included in the landmark.

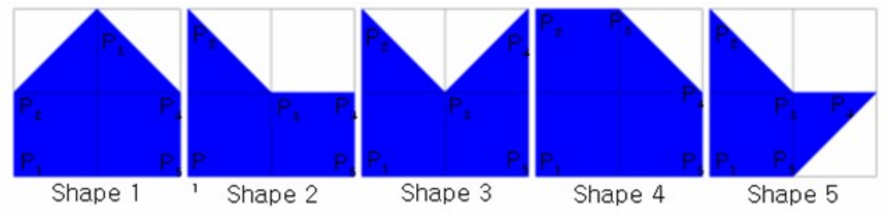


Fig. 2. Five Types of Basic Pentagons

Color is another factor to discriminate a landmark. We propose the colors of landmarks with different hues which are highly saturated so that the illumination change may not affect the differentiation. Note that the shapes and colors can be flexibly selected according to the working environment of AGV.

3 Image Processing and Self-localization

Assume that the internal parameters including camera constant (f) and scale factors of the camera are accurately calibrated before running in the environment. Then the camera is fixed to an AGV and its viewing direction is assumed to be oriented to the wall where landmarks are attached. During AGV running, the procedures to recognize the shape and color of a landmark and eventually to find self-location are as follows.

Step 1: Segment the current-captured frame to find a pentagon.
Step 2: Find vertexes of the pentagon.
Step 3: Recognize the shape and color of the pentagon to differentiate global location.
Step 4: Calibrate camera to determine the relative position and orientation of mobile object.

In Step 1, the intensity component from a color image is extracted and its edge map is taken by Canny edge detector. Every edge component whose number of pixels is greater than a threshold is labeled in the edge map. Again, the curvatures of pixels along each labeled edge component are calculated in order to verify whether it is on a boundary of pentagon or not. Note that there are five peaks of curvatures corresponding to the vertexes of a pentagon. Once the boundary of a pentagon is found, the pixel positions between the consecutive peaks are approximated by LMS (least mean square) fitting and a vertex of the pentagon is obtained by the intersection of two successive line equations. In Step 3, the invariant features of the pentagon are calculated and M uniform samples inside its boundary are taken to obtain the mode of hue distribution. Those features and the mode of hue are compared with the stored models to differentiate the shape and color of the pentagon. This gives the global location of a mobile object.

The camera model is defined by the equations. [4]

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix} \tag{2}$$

$$x_f = f_{ous}\frac{x_c}{z_c}, \qquad y_f = f_{ous}\frac{y_c}{z_c} \tag{3}$$

$$x_i = s_{caleX}x_f + x_{center}, \quad y_i = s_{caleY}y_f + y_{center} \tag{4}$$

Equation (2), (3), and (4) represents the absolute orientation of camera coordinates, the perspectives transform, and the mapping from the image plane to the pixel coordinates. There are two types of camera calibration. The one is internal camera calibration that is done before running. The camera constant (f), scale factors (S_{caleX} and S_{caleY}), and displacements (X_{ce} and Y_{ce}) are calculated using a reference grid pattern before running. Once those internal parameters are fixed, the external parameters associated with absolute transform are calculated using corresponding pairs of vertexes of a pentagon in Step 4. The parameters provides the translation amounts along x, y, and z axes and the orientation of camera including pan, tilt, and role. Among those parameters, the distance from a landmark and the pan angle is important to determine the heading direction of an AGV.

4 Experimental Results

In the experiment, we chose 15 shapes and 7 colors for making landmarks. That means there are 15x7 different landmarks. In 15 shapes, rotated or mirrored versions of the basic pentagons in Fig. 3 are also included. Corresponding hues of colors are given in Table 1. We took 2058 frames with JVC900Kr Camera, among which 1248 frames included at least one pentagon. For acquisition of images, Table 2 shows the performance of landmark detection by the image processing. Even though there are some missing frames that has at least one pentagon, the context of landmark sequence in the mobile environment can make it possible to determine the global position of AGV. Fig. 4 shows the clusters of pentagons in the invariant feature space. Note that the clusters are sufficiently far apart so that the nearest neighbor classification is well suited to recognize a pentagon. The internal parameters before running are estimated by Tsai's method, and listed in Table 3. In the running mode, we can estimate external parameters. Among them, the pan angle and the distance from the wall are important. The calibration by Tsai's method with five pairs of correspondence provided the results shown in Fig. 5 and Fig. 6. For the camera, the effective area in which the relative orientation and distance from wall

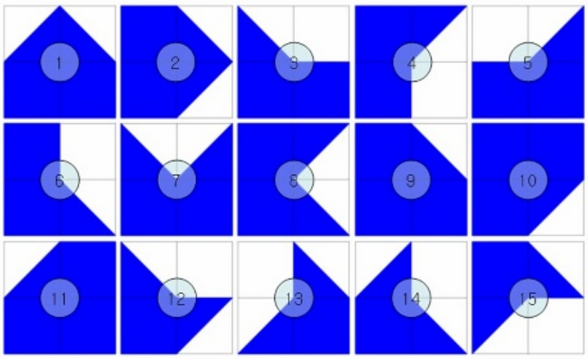


Fig. 3. Mirrored and Rotated Versions of Basic Pentagons

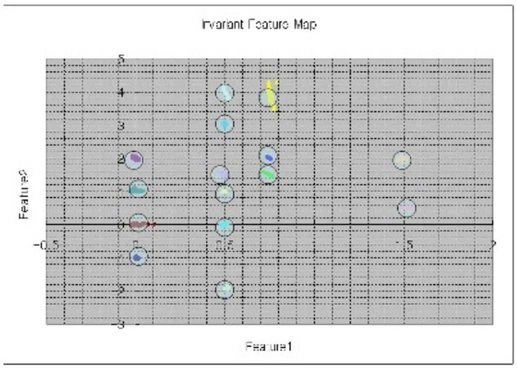


Fig. 4. Clusters of Pentagons in Invariant Feature Space

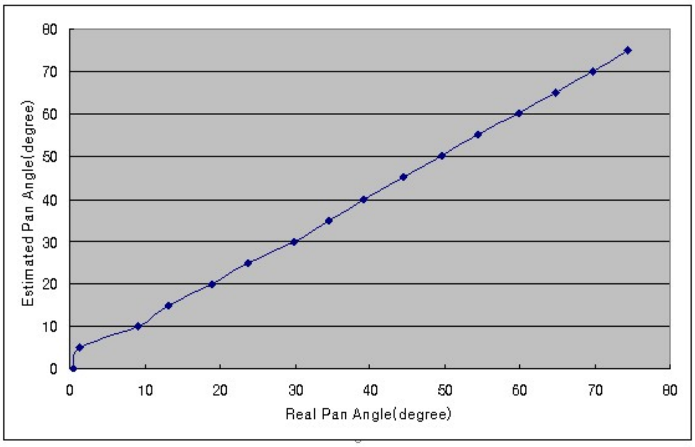


Fig. 5. Estimated amount of Pan(Ave. Difference 0.82 degree, Std Dev. of Difference 0.87 degree at Amount of Tilt = -0.5 degree, Role = 0, , t_x=240, t_y=-100, t_z=1600)

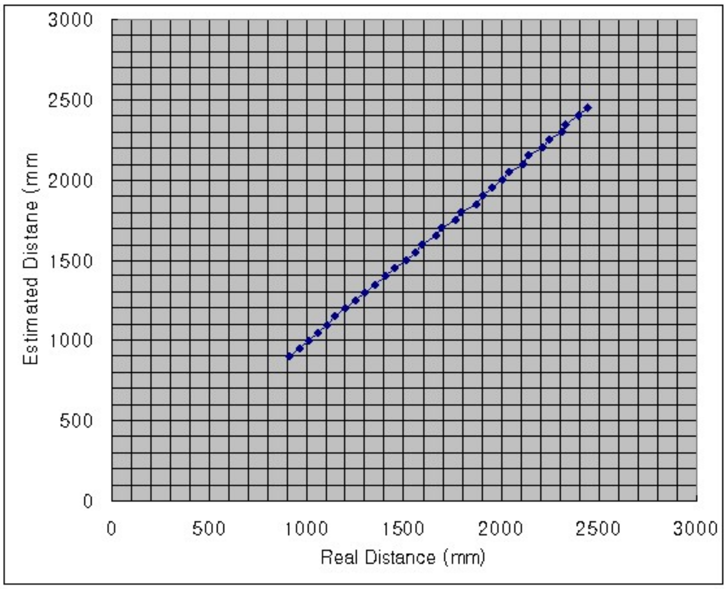


Fig. 6. Estimated Distance from Wall (Ave. Difference =7.75cm, Std. Dev. of Difference = 5.351cm at Tilt -Tilt=-7, Pan=-3, Role=0.5, X_0=30, Y_0=280)

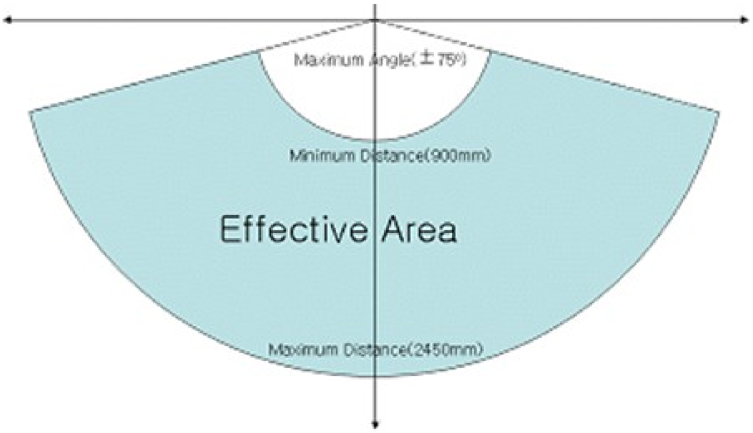


Fig. 7. Effective Area of Measurements of Pan and Distance

are stable enough given in Fig. 7. During the running, the calculated AGV positions are displayed, and Fig. 8 shows a part of locus of AGV. The time required to capture a frame and to estimate positions with relative angle was about 0.3 sec in Pentium 4 (2.4GHz).

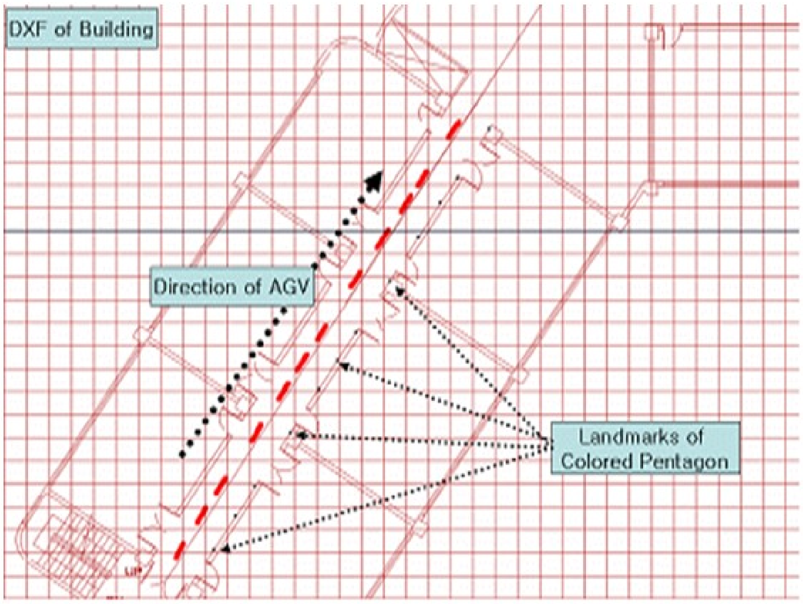


Fig. 8. Part of Self-localization in Corridor

Table 1. Hue of Colors of Pentagons

Colors	Minimum of Hue (degree)	Maximum of Hue (degree)
1	10	25
2	45	55
3	62	75
4	90	100
5	110	125
6	135	150
7	170	180

Table 2. Detection Performance of Landmarks expressed by a confusion matrix

Real / Detected	Frame with pentagon	Frame Without pentagon	Total
Frame with pentagon	1107	141	1248
Frame without pentagon	0	810	810

Table 3. Estimated Internal Parameters

	Eastimated	Error Ratio
Focal Length	$f_{ocus}\times S_{caleX}$: 977.55503 $f_{ocus}\times S_{caleY}$: 957.62681	X axis : 3.92925 Y axis : 3.68278
Principal point	X_{ce} : 342.40179 Y_{ce} : 210.58527	X axis : 3.50703 Y axis : 4.82543

5 Conclusion

This article proposes the colored pentagons as landmarks for self-localization of autonomous guided vehicle (AGV). This vision-based method can provide simple, robust and flexible localization of global position as well as relative orientation of AGV, because the invariance of colors and pentagons to the illumination changes and geometric distortions

Acknowledgements. This work was partially supported by the second stage of Brain Korea 21 Project in 2006.

References

1. Ml. R. Kabuka and A. Arenas : Position verification of a mobile robot using a standard pattern, IEEE J. Robotics Automat., vol RA-3, no. 6, pp. 505-516, Dec, 1987
2. Gijeong Jang, Sungho Kim, Wangheon Lee, Inso Kweon : Robust Self-localization of Mobile Robots using Artificial and Natural and marks In Proceeding 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation July 16-20, 2003, Kobe, Japan.
3. J. L. Mundy and Andrew Zisserman : Geometric Invariance in Computer Vision, MIT Press, Cambridge, Massachusetts, England, pp.14-22, 476-485, 1992.
4. R. Y. Tsai : A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-shelf TV cameras and lenses, IEEE Journal Robotics automation, Vol. RA-3, No. 4, pp 323-344, Aug. 1987.
5. M. Betke and L. Gurvits : Mobile Robot Localization Using Landmarks, IEEE Transactions on Robotics and Automation, Vol. 13, No 2, pp. 251-263, 1997.

An Automated System for Contact Lens Inspection

A.I. Bazin[1,2], T. Cole[2], B. Kett[2], and M.S. Nixon[1]

[1] School of Electronics and Computer Science, University of Southampton, Southampton, SO17 1BJ, United Kingdom
Tel.: +44 (023) 80592929; Fax: +44 (023) 80594498
{aib02r, msn}@ecs.soton.ac.uk
[2] Neusciences, Unit 2, Lulworth Business Centre, Nutwood Way, Totton, Southampton, SO40 3WW, United Kingdom
Tel.: +44 (023) 80664011; Fax: +44 (023) 80873707
{alex.bazin, trevor.cole, brian.kett}@neusciences.com

Abstract. This paper describes a novel method for the industrial inspection of ophthalmic contact lenses in a time constrained production line environment. We discuss the background to this problem, look at previous solutions and relevant allied work before describing our system. An overview of the system is given together with detailed descriptions of the algorithms used to perform the image processing, classification and inspection system. We conclude with a preliminary assessment of the system performance and discuss future work needed to complete the system.

1 Introduction

Industrial inspection is a vital part of the manufacturing process, especially in safety critical products such as medical devices. In this paper we describe a novel system for the inspection of ophthalmic contact lenses in a time constrained production line environment. Ophthalmic contact lenses are formed by injecting a monomer into an individual disposable hard plastic mould, formed to give the required lens curvature. Once the monomer had been cured in an oven, a manufacturing machine breaks open the moulds and separates the lens from the mould base. It is then transferred to an individual window for inspection before packaging (Fig. 1).

Due to the mechanical nature of the removal from the mould, together with occasional defects in the molding process, lenses are prone to a number of manufacturing defects. These include: bubbles within the monomer, splits or chips in the lens due to poor forming or damage in removal from the mould, attached monomer or rough edge due to poor removal from the mould, and contamination with particles of dust or debris. Since ophthalmic contact lenses are medical devices, the size and number of these defects must be strictly monitored and controlled. These inspection standards are laid down by government regulators and vary depending on the type and envisaged longevity of the lens.

We seek to produce a system that will perform automated inspection of ophthalmic contact lenses in a manufacturing environment. It is required to perform this inspection task at the accuracy level of a trained human operator whilst maintaining production line speeds. There have been a number of partial contact lens inspection or characterization systems described in the literature [1-3], as well as fault detection systems for other lens types [4]. However none of the systems described in the literature report the accurate fault detection and performance required for this system.

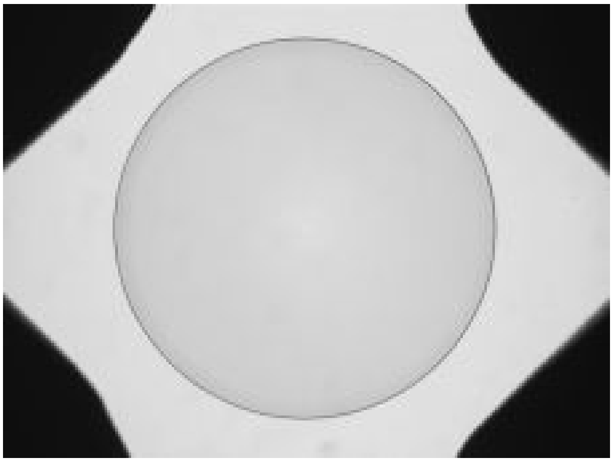


Fig. 1. An example lens image

This paper firstly provides an overview of the developed system including its interaction with the manufacturing equipment and human operators. This high level overview describes both the inspection system and allied control and monitoring software. We then describe in detail the methods used for processing the lens image, extracting relevant feature metrics, classifying fault types and comparing these classified features with the customer's inspection standards. The testing regime that has been implemented is discussed both with reference to the accuracy of the algorithms and the performance of the system as a whole.

We conclude by discussing the likely deployment of the system, work that remains to be completed and our work's wider relevance to industrial inspection.

2 System Overview

The system is divided into two separate processes designed to be run on separate machines. This allows monitoring and reporting to be separate from inspection; enabling remote working and multiple inspections to be running in parallel. The image processor contains modules to perform the full range of inspection activity and a separate process is instigated for each camera. On a single manufacturing line it is expected that there would be multiple cameras (and hence image processors) inspecting lenses in parallel. Provided there is sufficient processing power it is not necessary that this translates to one image processor per CPU requirement; this decision would be taken after fully considering the desired performance of the software and the hardware specification of the servers available. These multiple image processors are designed to be under the control of a single workstation process, run on a separate machine. The workstation process is responsible for set-up, display and reporting for the system. This workstation connects to the image processors remotely via TCP/IP and hence those deploying or monitoring the system do not need to be co-located with the manufacturing line.

This paper focuses primarily with the function of the image processor software; however we believe it is useful for the reader to understand the operation of the full system and its interaction with the wider manufacturing environment. A system diagram can be seen in Fig. 2.

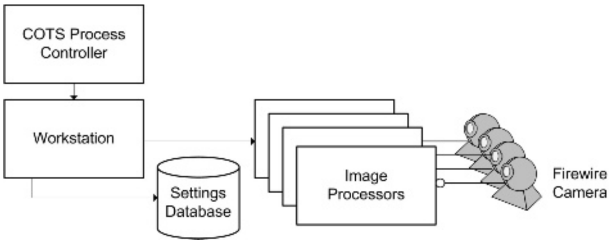


Fig. 2. The system diagram

Before a new 'batch' of lenses is to be inspected, the user must initialize the system. This involves firstly choosing which process modules are to be used, adjusting the settings for each module, loading the classifiers initialization files and creating the inspection standards for the lenses to be compared against. In the first instance these setups will be created by a supervisor and on subsequent runs the operator will simply select the appropriate setup for the type of lens on the manufacturing line.

Once the system is set up it may begin inspecting lenses. On the manufacturing line, once the lenses have been removed from their moulds but prior to being placed into packaging they pass below high resolution grey-scale cameras where an image of the lens is captured for inspection. The timing of this process is synchronized with the production process and is controlled by a Commercial Off The Shelf (COTS) process control device. This device tells the servers when a lens is under the camera and ready to be inspected, triggering the image processor to acquire the image and begin inspection. Whilst the image processor is inspecting the acquired image the process controller monitors the elapsed inspection time to avoid schedule overrun, should an overrun occur a signal is sent to abort the inspection of that image and reject the lens (in these cases the image would be queued for an offline inspection to diagnose the system fault that may have occurred). In the typical case where inspection is successfully completed within the stipulated time the process controller is informed of the pass/fail decision and the lens is either transferred to packaging or rejected as appropriate. The pass/fail decision as well as relevant statistics (feature counts and sizes etc) are passed via XML to the workstation for collation and reporting.

After a run is completed the operator can use the workstation to review the fault profile for that run and reprocess any images that timed out, in order to diagnose the system fault that caused this. During the run the workstation can be used to monitor the current and historic yield and identify recurring faults that may be indicative of a systemic manufacturing fault.

3 Modules

In order to maximize future flexibility the image processor is divided into separate modules. Each module typically implements one task or algorithm with a well defined set of inputs and outputs. This design methodology allows new techniques or additional functionality to be quickly added to the system. Each of the modules developed for the current system are described in this section.

3.1 Image and Lens Pre-processing

This module comprises of a number of algorithms which must be performed immediately after image acquisition to make the lens image ready for feature detection and further processing. Before processing of the lens occurs a check is made on the image where clear background is expected to be visible. The intensity is calculated and compared to standard values. If it diverges from expected values then this highlights either an obstructed view (i.e. debris on the window) or a failing illumination source or camera.

The initial processing step is calculating the centre of the lens. This is achieved by detecting the edge transition at spaced points around the lens. Once a number of points have been found then the centre may be converged upon using simple trigonometry. If no centre can be reliably found the software concludes that the lens is either not present or is suffering from some gross defect; in either case the steps described in below and in sections 3.2 through 3.6 are not performed and instead the algorithm described in section 3.7 is invoked.

Having detected an accurate centre for the lens it is now necessary to fit appropriate ellipse parameters to describe the edge. Initially we considered using an active contour approach [5], however this proved overly complex for the regular shape of the lens. In initial tests direct fitting [6] and Constrained Hough Transforms [7, 8] were also judged computationally inefficient in our constrained environment with predictable lens shapes. The method found to be both sufficiently accurate and efficient was a the Randomized Hough Transform which has been variously described [9, 10]. Since the normal size and shape of the lens will be known for any given batch of lenses and that the centre has already been accurately calculated, it is possible to strongly constrain the RHT to very rapidly converge on accurate parameters.

Once the centre and ellipse parameters have been accurately estimated, the real outer and inner edges of the lens are extracted. This is achieved by finding the transition from the darker edge to the lighter inner lens (the inner edge) and from the darker edge to the much lighter background (the outer edge).

3.2 Surface Feature Detection

Our system defines the surface area as a circular region covering the centre 90% of the lens. It is in this region that surface features are searched for, the special case of a feature extending between the surface and edge region is dealt with in section 3.3.

To find surface features of interest a modified Canny operator [11] with a 5x5 window is run over the entire surface region. In order to prevent small gaps creating multiple features out of a single poorly defined feature the hysteresis thresholding stage is allowed to consider pixels in a 5x5 neighborhood rather than simply adjoining pixels. The Canny operator produces a binary image of feature points that will be of interest to us.

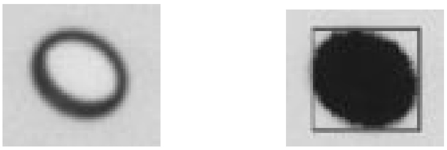


Fig. 3. A bubble hole in the lens monomer before and after extraction

Once the Canny operator has been used; spatially separate features are extracted for feature description. Starting with the uppermost pixel in the surface region we scan left to right working progressively downwards until we find a pixel that has been marked by the Canny operator as a feature pixel. This then becomes a seed point for a new feature. Any feature points within a 5x5 neighborhood of this pixel are also added as seed points for the feature and their neighborhood is examined. Once all adjoining pixels have been checked the scan for feature pixels continues and when a new feature pixel is found the extraction of neighbors is repeated to yield another feature. This is repeated until all surface features have been extracted into separate array lists containing the pixel locations. An example of a bubble in the monomer and the extracted feature after processing described in section 3.4 can be seen in Fig. 3.

3.3 Edge Feature Extraction

The edge region of the lens is defined as an annulus covering the outer 10% of the lens and for our purposes we also consider a small region outside of the outer edge to search for debris attached to the lens.

Using the extracted ellipse parameters we 'unwrap' the annulus to form a rectangular image. Having formed the unwrapped image we then perform checks along the outer and inner edge to find small edge faults. These tests look for significant deviation in the spacing between the outer and inner edges, deviation of the edge from the fitted ellipse and variations in intensity.

After performing heuristic checks for faults features in the edge band are extracted in the same manner as described in section 3.2 with one important exception. If a feature extends into the surface the edge feature extractor searches the interface region for connecting features and merges these into one. Fig. 4 shows an edge fault in the original image and the same edge fault after extraction in the unwrapped edge image.

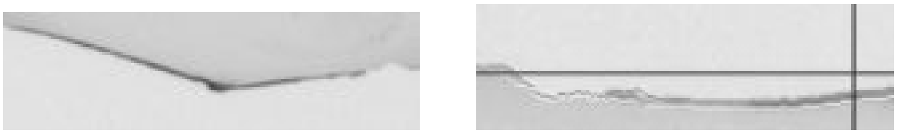


Fig. 4. An edge fault before and after unwrapping and extraction

3.4 Feature Description

Once we have a set of features, all stored as ordered array lists of pixels we process each feature to extract mathematical descriptors for classification.

We first extract the perimeter of the feature (i.e. identify those pixels that fully enclose the feature). We achieve this by starting with the upper left pixel of the feature and progressing in a clockwise direction to find the next neighboring pixel. By structuring our neighbor search in a clockwise direction we can guarantee that we always find the outermost neighboring pixel.

Having extracted the perimeter of the feature we then fill it for use in further mathematical descriptors. The fill is performed by working clockwise and filling between the perimeter in either an upwards or downward direction as appropriate. Checks are made to ensure the perimeter is not a single line at this point to ensure that the fill does not escape the feature.

Given a collection of pixels representing the perimeter and filled feature we can then extract mathematical measures of the shape for classification. We firstly calculate gross shape measures: perimeter length, area, maximum chord, minimum chord, dispersion and compactness [12]. Compactness is a measure of the perimeter relative to the area and dispersion is the ratio of the largest circle enclosed by the feature to the smallest circle enclosing the feature. More complex measures are produced by calculating the first four rotation invariant moments [13]. These moments are invariant to position, size and rotation.

We also extract information about the grey-scale intensity of the feature; mean intensity and standard deviation. Additional features in the edge region have Boolean information appended to describe their position in the region and whether they extend outside of the lens or into the surface region.

3.5 Feature Classification

Once we have extracted mathematical information to describe our feature we then must classify which fault type the feature most closely resembles. To simplify this we split the features into three types based on their location within the lens: surface feature, edge feature, and surface feature in edge band. We do this in order to remove implausible classification possibilities from the set of outcomes and because the surface and edge features have different feature vectors due to additional Boolean tests on the edge.

A probabilistic classifier is implemented for the two groups of surface features. This type of classifier is used to provide additional feedback to the system for further analysis and diagnostics. Probabilistic classifiers output a decision confidence in addition to their decision; this is useful since it provides operators with feedback as to how well the classifier is performing and whether the decision is good enough to be relied upon.

We implement a Bayesian probabilistic classifier based on logistic functions [14]. This implementation was chosen because of its high performance on continuous data. For our formulation we assume that the classes form a complete and mutually exclusive set, and currently assume that each class is equiprobable (though this is to be tuned once sufficient evidence is obtained).

The edge classifier is implemented as a C4.5 decision tree [15] trained to identify those faults that may be found in the edge region and other non-fault artifacts that may also be detected. A different implementation to the surface classifier was used due to the Boolean values in the edge feature vectors, making a Bayesian classifier unsuitable. The classifier is implemented as a java bean from Neuscience's NeuJDesk range, and is trained offline using hand labeled faults that have been extracted in the manner described in sections 3.1 through 3.4.

3.6 Inspection Standards Comparison

As discussed in section 1 there exist strict criteria for the size and number of defects that may be present in any ophthalmic contact lens and as with most other medical regulations the outcome of these comparisons must be deterministic, strictly adhered to and carefully documented.

Having determined the fault type of each feature (section 3.5) and the size of the feature (section 3.4) we may then compare each feature against the predefined inspection standard for the lens type under examination. Every feature is recorded according to whether it causes an outright fail, whether it could contribute to a cumulative fail, or whether it is of a type or size to not be significant for our decision.

Once every feature has been compared against the standard, the whole standard is checked to see if any failures have been recorded; either cumulative or outright. If there are one or more failures then the COTS process controller is instructed to reject the lens and the major failure mode is recorded; otherwise then the COTS process controller is instructed to pass the lens for packaging and an entry of 'no failure' is entered into the system logs.

We have also made it possible that inspections against multiple standards are possible for regulatory or commercial reasons, however only the primary standard is used to instruct the COTS process controller.

3.7 Gross Fault Detection

Should a valid lens centre or ellipse not be detected as described in section 3.1, rather than processing the lens in a way which is likely to fail in a catastrophic manner, we instead perform a high level examination of the image in order to determine one of three gross failure modes: no lens present, lens fragment, shattered lens. There is the possibility that large debris could have obscured the window though this would likely cause the illumination check to fail. An example of a lens suffering from a gross failure can be seen in Fig. 5.

To perform this check we accumulate pixels over the entire image into three 'bins'. These are: pixels of about background intensity, pixels of about lens surface intensity, and pixels of about lens edge intensity. By comparing these with the number expected of a complete lens we can judge how much of a lens is present. Furthermore by comparing the ratio of edge intensity to surface intensity pixels we can determine the extent to the deformation of the lens.

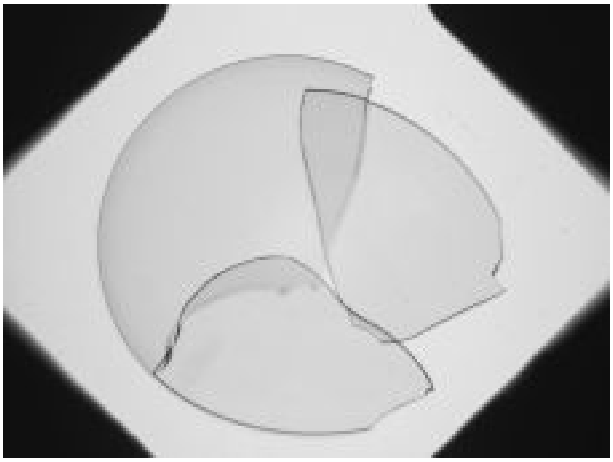


Fig. 5. A shattered lens

4 Testing

In evaluating the system against the requirements of the project we have considered a number of tests both at the module and system level.

4.1 Module Tests

We have tested each module sequentially and compared the outputs with expert opinion and the performance of other systems. In the pre-processing stage we compared the extracted centre coordinates and ellipse parameters with hand marked lenses to ensure pixel level accuracy in the extraction. For feature extraction steps we have consulted widely with experts in the field to ensure that the system detects all features and artifacts that are detected by a human expert.

The classifiers have been trained and tested on separate hand-labeled features and perform at a very high level of accuracy. We have also ensured that the feature extraction and inspection standards processes perform as intended by careful comparison with reference implantations.

4.2 System Tests

Having ensured that all system components are performing as expected we have performed tests on the whole system to ensure that timing and yields are as expected. In initial tests on a small number of images (a few hundred) we can achieve correct reject/accept decisions on 100% of lenses including correct largest failure mode. Current trials indicate that sub one second processing times are achievable on standard 1GHz, 512MB Windows 2003 Enterprise Server and there is scope for further compiler optimization. The use of comparable exhaustive established techniques for feature detection would fail to meet these time constraints.

5 Conclusions

In this paper we have described a novel method for the industrial inspection of ophthalmic contact lenses in a time constrained production line environment. In describing this system we have discussed the requirement for a fast an accurate inspection system for fault detection in regulated medical devices. We have given an overview of the system including interfaces to other systems and with operators. We also have described in detail the modules that comprise the inspection system and the tests that these modules have undergone. Finally we briefly describe the full system tests we have performed to establish that our system meets the specifications laid down.

There still exists work to be completed on this system, particularly in interfaces to the manufacturing controller, logging and reporting and the user interface. We also wish to continue our work on the characterization of the system across a much greater numbers of lenses and on more typical server platforms.

This work has applicability to a wider field than inspection of ophthalmic contact lenses; there are many products that need rapid accurate fault detection with similar fault profiles to those seen in this work. This is especially relevant to those situations where immediate feedback of such results can be used to adjust process parameters. Additionally the processes developed here may find uses in non industrial inspection applications, such as pathological screening applications and object recognition systems.

Acknowledgements

The authors gratefully acknowledge the support of CooperVision, Kirk Leppard and the Engineering and Physical Research Council, UK.

References

1. Elliott, C.J. *Automatic optical measurement of contact lenses*. in *Proc. SPIE - Automatic Optical Inspection*. 1986. Innsbruck, Austria.
2. Pladellorens, J.M., et al. *Analysis and characterization of surface defects in ophthalmic lenses*. in *Proc. SPIE - Surface Scattering and Diffraction for Advanced Metrology II*. 2002. Seattle, WA, United States.
3. Hobbs, P.C.D. *Ideas for fast and cheap object capture*. in *Proc. SPIE - Three-Dimensional Imaging, Optical Metrology, and Inspection IV*. 1998. Boston, MA, USA.
4. Cho, J.H., M.W. Cho, and M.K. Kim, *Computer-aided design, manufacturing and inspection system integration for optical lens production.* Int'l Journal of Production Research, 2002. **40**(16): p. 4271-83.
5. Gunn, S.R. and M.S. Nixon, *A robust snake implementation; a dual active contour.* IEEE Trans. PAMI, 1997. **19**(1): p. 63-8.
6. Fitzgibbon, A., M. Pilu, and R.B. Fisher, *Direct least square fitting of ellipses.* IEEE Trans. PAMI, 1999. **21**(5): p. 476-80.
7. Olson, C.E., *Constrained Hough transforms for curve detection.* Computer Vision and Image Understanding, 1999. **73**(3): p. 329-45.

8. Xie, Y. and Q. Ji. *A new efficient ellipse detection method*. in *Proc. 16th Int'l Conf. Pattern Recognition*. 2002. Quebec City, Que., Canada.
9. Cheng, Z. and Y. Liu. *Efficient technique for ellipse detection using restricted randomized Hough transform*. in *Proc. Int'l Conf. Information Technology: Coding and Computing*. 2004. Las Vegas, NV, USA.
10. McLaughlin, R.A., *Randomized Hough transform: improved ellipse detection with comparison.* Pattern Recognition Letters, 1998. **19**(3-4): p. 299-305.
11. Canny, J., *A computational approach to edge detection.* IEEE Trans. PAMI, 1986. **8**(6): p. 679-98.
12. Nixon, M.S. and A.S. Aguardo, *Feature Extraction & Image Processing*. 2002: Newnes.
13. Hu, M.K., *Visual Pattern Recognition by Moment Invariants.* IRE Trans. Information Theory, 1962. **8**: p. 179-187.
14. Bazin, A.I. and M.S. Nixon. *Gait verification using probabilistic methods*. in *Proc. 7th IEEE Workshop on Applications of Computer Vision*. 2005. Breckenridge, CO, USA.
15. Quinlan, J.R., *Simplifying decision trees.* Int'l Journal of Man-Machine Studies, 1987. **27**(3): p. 221-34.

Efficient Motion Search in Large Motion Capture Databases

Yi Lin

University of Waterloo

Abstract. Large human motion databases contain variants of natural motions that are valuable for animation generation and synthesis. But retrieving visually similar motions is still a difficult and time-consuming problem. This paper provides methods for identifying visually and numerically similar motions in a large database given a query of motion segment. We propose an efficient indexing strategy that represents the motions compactly through a preprocessing. This representation scales down the range of searching the database. Motions in this range are possible candidates of the final matches. For detailed comparisons between the query and the candidates, we propose an algorithm that compares the motions' curves swiftly. Our methods can apply to large human motion databases and achieve high performance and accuracy compared with previous work. We present experimental results on testing a database of about 2.9 million frames, or about 27 hours of motions played at 30 Hz.

1 Introduction

Recently, large motion capture databases have become commonplace due to real-world projects requiring expressive character motions. These databases contain many different kinds of actions and any action can have many variants. Theoretically, it seems that we do not need to capture motions redundantly and that we could create realistic motions simply by connecting the required motions in the database. This might be feasible if only we could find appropriate motions fast enough. This is not as easy as it looks, especially with large databases.

The main reason is that the currently used retrieval strategy involves hand-annotating each motion with a descriptive label. The annotations are often far from describing the motion clearly. For example, a label "punch" may represent many different motions. Different annotations may also be unable to reflect the relations between motions. For instance, a "punch" may related closely with a "dodge a counter-blow". A real world user often has to scan the database, examine every possible candidate motion, and crop the frames of interest. Manually searching a large database is an insufferably time-consuming job.

There exist approaches that allow the query to be a short motion segment, and that automatically retrieve all motion segments in the database containing parts or aspects similar to the query. The basic idea of these kinds of approaches is that the database is preprocessed using an indexing strategy for fast retrieval.Various indexing strategies have been developed in the past few years, such as constructing match webs [1], partitioning motions using geometric features [2], and clustering frames using index trees [3]. However, there is always a tradeoff between accuracy and efficiency for this problem. Previous work either focuses more on "accuracy" with less "efficiency" consideration, or vice versa.

In this paper, we present efficient indexing and retrieval methods for searching large databases given a query of a motion segment. The indexing method limits the range of data access and detailed comparisons for retrieval. This range contains a set of candidate motions with similar geometric features. In this set, the retrieval method can search for best matches swiftly. In this way, the methods can achieve high accuracy and efficiency.

The paper is organized as follows. The reminder of this section presents an overview of our methods and a summary of contributions. In Section 2 we discuss related work. In Section 3 we details of our indexing method. In Section 4 we introduce our motion matching algorithm. In Section 5, we present experimental results. Finally, in Section 6 we conclude this paper by discussing the advantages and limitations of our methods and providing directions for future work.

1.1 Overview

1. **Geometric feature based indexing.** Since we aim at large databases (in our project, we use the CMU motion capture database [4], whose size is about 2.2 gigabytes), it is not realistic to load the whole database into the main memory, especially for high-performance applications. Multiple accesses of the hard disk will decrease the performance greatly. This problem has seldom been considered by previous work. We propose an indexing method which preprocesses the database by partitioning motions into segments and clustering the similar motions based on a general class of geometric features. This index structure will accelerate retrieval by rapidly selecting a small set of candidate motions from the full database.
2. **Efficient motion curve matching.** After the candidate set has been determined, we compare the query and the motion in the set by high-performance applications calculating their similarity distance. The typical matching method is dynamic time warping (DTW), which is expensive in computation requiring $O(mn)$ time (where m is the number of frames of the query, n is the number of frames of the motion, and $m \leq n$). DTW is used in most of the current motion retrieval systems. We propose an efficient motion curve matching which only requires $O(k)$ time (where $k \leq m$). This method can find similar motion segments with different length and frequency as the query.
3. **Both visual similarity and numerical similarity.** Visually similar character motions may be numerically dissimilar because corresponding frames may have very different joint orientations and angular velocities. Traditional algorithms implicitly equate numerical similarity with visual similarity, and they are often unable to distinguish motions that are unrelated from those that are different versions of the same kind of action. Our indexing method considers geometric features and clusters visually similar motions into a candidate set for detailed comparisons of matching.

2 Related Work

Motion capture data has only recently become available publicly. However, efficient retrieval techniques in multimedia databases, such as text, image, music and video data have been researched for many years and a vast literature exists [5,6,7]. Some of these methods can be extended to the motion retrieval problem, such as indexing by hashing tables or trees, content-based retrievals and motion matching using DTW.

DTW is the typical method to compare two time series. DTW finds "legal" paths with minimal costs in the distance matrix. DTW can achieve highly numerically accurate similarity but its efficiency is low. In addition, the numerical comparison in the distance function may not reflect the real visual similarity. One improvement is to use the found motions as new queries to find more similar motions [1]. In this way, visually similar motions can be found iteratively. Obviously, this method requires more computation time.

In 1994, Faloutsos et al. proposed the GEMINI framework for motion sequence retrieval followed by many researchers [8]. The basic idea of the GEMINI framework is the following. First, it approximates the high-dimensional data to a low dimensional representation using dimension reduction methods. This representation can be expressed as a Fourier transform [9], wavelet transform [10], average values in adjacent windows [11], or bounding boxes [12]. Then a distance metric is defined over this approximation. The low-dimensional data is often stored in a spatial data structure, like an R-tree [13]. This low-dimensional expression accelerates processing, but sacrifices accuracy.

Some researchers have investigated methods without dimension reduction. For frame comparisons, the distance function is based on the L_p norm, which may vary with different applications, as do the actual values that are compared. For example, Lee et al. [14] use joint orientations and velocities, which are common parameters. A similar hybrid method proposed by Arikan and Forsyth [15] involves joint accelerations. Usually, a weight parameter is given to each bone to specify influence of the bone on the whole pose. Although this kinds of distance functions based on the entire set of data causes low efficiency, it has become the basic universal metric function of frame comparison. What is not agreed on is how we should set the weights. It is obvious that some bones are more important than others, but how to specify the weights is still under argument. Wang and Bodenheimer [16] optimize the weights based on the cost metric used in Lee et al.'s work [14]. Our matching method is similar to this kind of methods. The distance cost is the weighted sum of the distances between motion curve peaks, instead of frames.

Recently, researchers begin to consider geometric features as distance metrics directly. One example is Muller et al. [2] who presented a system in which user-specified geometric features are a part of the query with the motion itself. The indexing strategy is also based on these features. This method is difficult to apply to complex feature combinations. Its query mode requires more labor for the user. If the geometric features change, the database has to be re-indexed completely. In our paper, we use a general class of geometric features to indexing the database automatically, which does not require the user to input the geometric features.

3 Geometric Feature Based Indexing

We construct an index tree with the levels as bones and the branches as the geometric features of the bones. For most motions, only some bones dominate the pose, such as the back, the arms and the legs. So we only consider the geometric features of these important bones, which we called *featured bones*. In this way, the height of the index tree that we are going to build can be shortened greatly, and the searching efficiency can be improved. Unlike the Boolean geometric features in [2], which are coarse and express only a single geometric aspect, ours is a class of general features. The featured bones in our prototype system are: torso, left upper leg, right upper leg, left upper arm and right upper arm. They consist of five levels of the tree from top to bottom. We choose these bones because we notice that these bones influence the visual similarity most.

The basic idea of building the index tree is to partition motion into segments based on geometric features and to insert "similar" segments into a leaf of the index tree. Each leaf is a cluster of motion segments with the same feature code. Each branch represents a geometric feature of one bone using a feature code. We define a feature coding function to represent a frame as

$$f : M[j] \leftarrow \{0, 1, 2\}^k,$$

where k is the number of the featured bones and f is our feature function. We suppose the signed distance from p to the plane (p_1, p_2, p_3) is $d(p_1, p_2, p_3; p)$, in short d. Our feature definition is

$$f(p_1, p_2, p_3 : p_4) := \begin{cases} 0, d \in [d_{\min}, d_{\max}] \\ 1, d > d_{\max} \\ 2, d < d_{\min} \end{cases}, \tag{1}$$

where $d_{\min}$ and d_{max} are pre-defined threshold values. Table 1 lists current features used in our prototype system.

Each leaf node is associated with a subset which contains promising similar motion segments with the same feature code. All these subsets constitute the whole motion database. The leaves of the tree are shown in Figure 1. Similar motion segments are clustered in the motion set. The sequence between motion segments are linked by bidirectional pointers. Each leaf node in the motion index tree contains the index structures that we called inodes. An inode structure is given by: *inode = code, file, start frame #, end frame #, link to previous inode, link to next inode.*

3.1 Motion Curve Matching

The expensive dynamic time warping method prohibits high-performance and real-time applications. In order to reduce computation time, we may ask: do we need to compare each pair of corresponding frames of the two segments? In fact, this problem has been investigated well in the bioinformatics area to find approximate repeats or homologies in DNA sequences. Many sophisticated

Table 1. Our general geometric features

Feature set	Explanations
f_{torso}	torso stands up, leans forward or bends backward
f_{lleg}	left leg stands, move forward or back backward
f_{rleg}	right leg stands, move forward or back backward
f_{larm}	left arm drops down, move forward or back backward
f_{rarm}	left arm stands, move forward or back backward

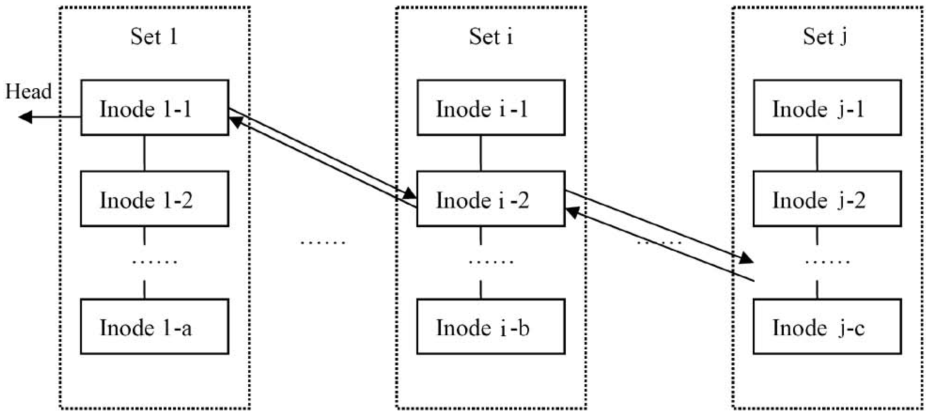


Fig. 1. Similar motion segments are clustered in the motion set. The motion segments are linked by bidirectional pointers.

algorithms haven been proposed, e.g., [17,18]. Some bioinformatics algorithms improve the time to $O(kn)$ ($k << m$) by using matching patterns. Unlike DNA sequences that are purely discrete, motion sequences are essentially continuous. So we can not apply these patterns directly to motion comparisons. However, inspired by this idea, we develop a curve peak pattern matching method. Since we know all the matching candidates should have the same feature sequence, we propose an efficient curve matching algorithm which only compare the peaks of the curves.

We suppose the query motion is $Q[1:m]$ and the one matching candidate is $M_i[1:n]$ $(m \leq n)$. The mean frames of Q and M_i are mean$(Q) = \sum_{j=1}^{k} Q[j]$ and mean$(M_i) = \sum_{j=1}^{k} M_i[j]$. We then compute the variances of the frames. We define the peaks as those variances that are the maximums or minimums in a continuous range of values that are greater or smaller than the mean value. This method is illustrated in Figure 2. Supposing the peak sequences of the query and one matching candidate are $x_1, x_2, \cdots, x_m$ and $y_1, y_2, \cdots, y_m, \cdots, y_n$, the similarity cost of the sequences is:

$$cost = \sum_{j=1}^{m} w_j |x_j - y_j| e^{c|sign(x_i) - sign(y_i)|},$$

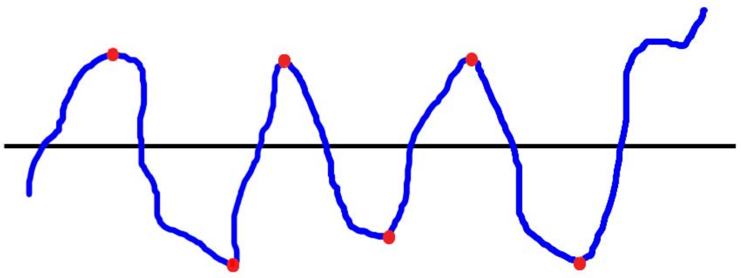


Fig. 2. The motion curve of one bone's parameter is shows as the blue curve. The black line is the mean value. The red points are the peaks.

Table 2. The performance of indexing four motion segments

Motion description	**# Segment**	**# Frame**	**Time (s)**
Walk	3	120	0.137
Run jump	9	100	0.297
Run	7	90	0.159
Kick a ball	12	150	0.197

where w_b is the weight assigned to a bone, B is the number of bones, and c is a constant. The matching candidate having the least cost is the best match.

This method can reduce computation time greatly. It does not consider the length and the frequency of the query and the candidates. It can find matching segments with the same style but different lengths and frequencies as the query. In our experiments, we use a threshold value d_l to select those matches whose lengths are in a range of d_l distant from the length of the query. In this way, we constrain the matches in the frequency field.

4 Experiments and Results

We implemented our indexing and matching algorithms in Java and tested them on a database containing 1460 AMC files, about 2.9 million frames (about 26 hours sampled at 30 Hz.) The experiments were run on a 2.6GHz Pentium 4 with 512MB of main memory.

Indexing the whole database took 1,591.864 seconds (about 26.53 minutes) clustering 69,372 motion segments and 139 sets. We use four motion segments as queries, which involve motions of walk, run jump, run and kick a ball. The performance of indexing these queries is shown in Table 2.

The performance of matching is shown in Table 3. We scale d_l from 0 to 110. We use a segment in the database as the query to test the exact matching. From the table, we can see that if d_l is too small (≤ 10), the method do not find the exact match. We examine the first ten best matches found by the method and

Table 3. The performance of matching

d_l	**# Segment**	**# Frame**	**Indexing (s)**	**Matching (s)**	**Loading (s)**	**# Visual matches**	**# Exact matches**
0	0	0	0.029	0	0	0	No
5	8	959	0.105	0.098	0.581	0	No
10	25	2,978	0.171	0.165	1.111	2	Yes
20	60	7,058	0.183	0.231	3.531	2	Yes
30	93	10,752	0.230	0.332	4.809	3	Yes
40	125	14,242	0.194	0.380	9.717	3	Yes
50	164	17,597	0.283	0.438	15.518	4	Yes
70	270	25,249	0.334	0.563	23.946	6	Yes
90	356	29,903	0.362	0.775	31.019	7	Yes
110	383	31,573	0.509	0.669	35.396	7	Yes

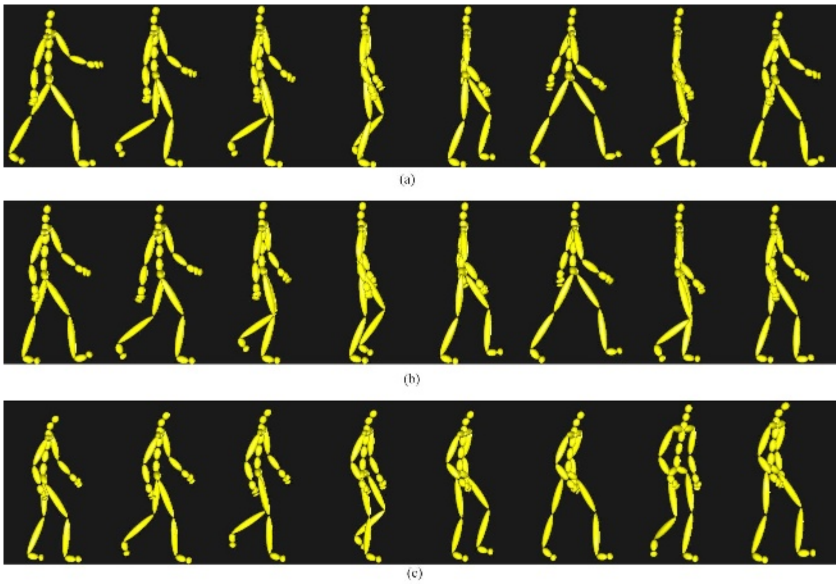


Fig. 3. (a) Samples of a query of walking (120 frames) (b) The best match (c) the 5th match

call those that are visually similar to the query the *visual matches*. We see that with dl increases, there are more visual matches in the top ten. However, the processing time increases too. Fortunately, all the processing time is within one second. The time-consuming part is loading data from the hard disk to the main memory. This is a hardware issue and unable to be improved by software.

The matching results are shown in Figure 3, 4, 5 and 6. Here we use $d_l = 50$. In Figure 3, we can see the motion of best match is very close to the query, while

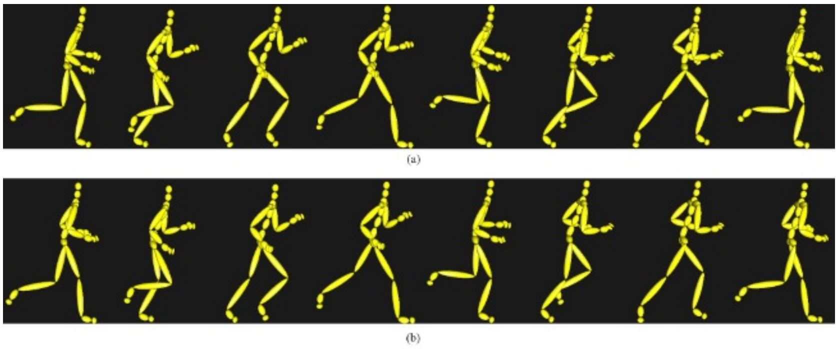


Fig. 4. (a) Samples of a query of running (90 frames) (b) The best match

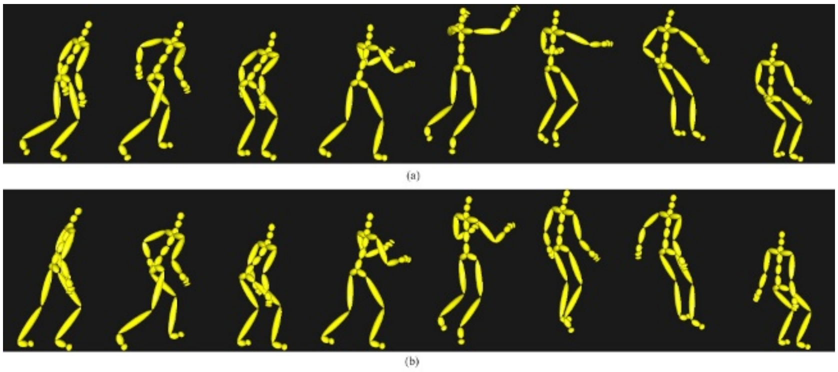


Fig. 5. (a) Samples of a query of running and jumping (100 frames) (b) The best match

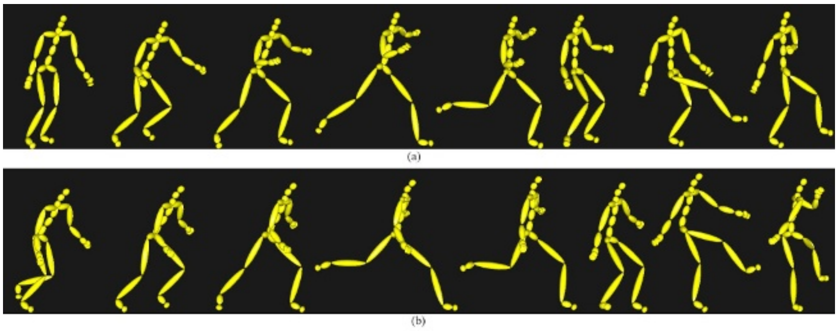


Fig. 6. (a) Samples of a query of kicking a ball (150 frames) (b) The best match

the difference from the 5th match and the query is quite obvious. But they are still visually similar walking.

5 Conclusion and Future Work

In this paper we focus on providing solutions for efficiently searching a large human motion database. Our solutions achieve high accuracy and efficiency. The resulting motions of the matches are visually similar to the query motion. Compared with previous work, we use the geometric feature based indexing tree to limit the range of hard disk access and data comparison. Based on this indexing strategy, our matching algorithm swiftly compare two motion segments by comparing the peaks of the motion curves. It achieves constant computation time.

The most brittle part of our system is the geometric feature selection. Since different motion styles have different significant features, our general class of geometric features may not work for all kinds of motions. If the features are not selected appropriately, the matching results will be visually unsatisfied. In Table 3, we can see that some of found matches are not visual matches in the top ten matches.This is mainly because of the feature selection. However, if the user only wants several matches, our methods work can provide immediate results.

Future Work: We will further experiment feature selection and try to abstract better general features. Motion data is high dimensional data and the channels of data are independent from each other, which is very suitable for GPU parallel processing. Our next step is to investigate the algorithms with GPU acceleration. There are applications that could use our work. One example is the animation of crowds. A lot of characters move simultaneously. It requires high performance of computation. Also, real-time user interfaces of animation could be built on top of our system if the performance is sufficient.

References

1. Kovar, L., Gleicher, M.: Automated extraction and parameterization of motions in large data sets. In: Proceedings of ACM SIGGRAPH 2004. (2004) 559–568
2. Muller, M., Roder, T., Glausen, M.: Efficient content-based retrieval of motion capture data. In: Proceedings of ACM SIGGRAPH 2005. (2005) 677–685
3. Liu, F., Zhuan, Y., Wu, F., Pan, Y.: 3d motion retrieval with motion index tree. Computer Vision and Image Understanding **92** (2003) 265–284
4. Graphics Lab, Carnegie-Mellon University: (Carnegie-Mellon MoCap Database) http://mocap.cs.cmu.edu.
5. Witten, I., Moffat, A., Bell, T.: Managing Gigabytes. Morgan Kaufmann Publishers (1999)
6. Bakker, E., Huang, T., Lew, M., Sebe, N., Zhou, X.: Eds. 2003Proceedings of 2nd International Conference Image and Video Retrieval, CIVR 2003. Volume 2728 of LNCS. Springer, Urbana-Champaign, IL, USA (2003)
7. Clausen, M., Kurth, F.: A unified approach to content-based and fault tolerant music recognition. IEEE Transactions on Multimedia **6** (2004) 717–731

8. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast subsequence matching in time-series databases. In: Proceedings of 1994 ACM SIGMOD International Conference on Management of Data. (1994) 419–429
9. Agrawal, R., Faloutsos, C., Swami, A.: Efficient similarity search in sequence databases. In: Proceedings of the 4th International Conference on Foundations of Data Organizations and Algorithms (FODO), Springer Verlag (1993) 69–84
10. Chan, K., Fu, W.: Efficient time series matching by wavelets. In: Proceedings of the 15th IEEE International Conference on Data Engineering. (1999) 126–133
11. Keogh, E., Chakrabarti, K., Mehrotra, S., Pazzani, M.: Locally adaptive dimensionality reduction for indexing large time series databases. In: Proseedings of 1994 ACM SIGMOD International Conference on Management of Data. (2001) 151–162
12. Vlachos, M., Hadjieleftheriou, M., Gunopulos, D., Keogh, E.: Indexing multidimensional time-series with support for multiple distance measures. In: Proseedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. (2003) 216–225
13. Guttman, A.: R-trees: a dynamic index structure for spatial searching. In: Proseedings of 1994 ACM SIGMOD International Conference on Management of Data. (1984) 47–57
14. Lee, J., Chai, J., Reitsma, P., Hodgins, J., Pollard, N.: Interactive control of avatars animated with human motion data. In: Proceedings of ACM SIGGRAPH 2002. (2002) 491–500
15. Arikan, O., Forsyth, D.A.: Interactive motion generation from examples. In: Proceedings of ACM SIGGRAPH 2002. (2002) 483–490
16. Wang, J., Bodenheimer, B.: An evaluation of a cost metric for selecting transitions between motion segments. In: Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2003. (2003) 232–238
17. Brejova, B., Brown, D., Vinar, T.: Vector seeds: An extension to spaced seeds. Journal of Computer and System Sciences **70** (2005) 364–380
18. Ma, B., Tromp, J., Li, M.: Patternhunter: faster and more sensitive homology search. Bioinformatics **18** (2002) 440–445

Real-Time Rendering of Light Shafts on GPU

Shuyi Chen, Sheng Li, and Guoping Wang

Lab. of HCI & Multimedia, School of Electronics Engineering and Computer Science, Peking University, China, 100871
{lisheng, wgp}@graphics.pku.edu.cn

Abstract. In the past, it is difficult to simulate light shafts effect in real-time. One major reason is the high computational expense to perform the physically-accurate computation of atmosphere scattering. Another is due to the limitation of computer resource, especially lack of power and programmability in the graphic hardware. Recently, with the advent of more powerful graphic card in standard PC platform and the development of programmable stages in the graphic pipeline, a lot of computational expensive algorithms are made available in modern commercial games. In this paper, we propose a novel method of rendering light shafts with atmospheric scattering based on actual physical phenomena. The proposed method utilizes hardware frame buffer object and a mesh refinement pattern to achieve photorealistic effect at high frame rate.

1 Introduction

Realistic image synthesis is one of the most important research subjects in computer graphics. To create physically-accurate realistic image, the effect of the scattering and absorption of light due to atmospheric particles is one of the most important elements to be taken into consideration. This effect mainly includes sunlight, skylight, aerial perspective and light beams caused by headlights of automobiles, street lamps, studio spotlights, and light passing through stained glass windows. During the past, those effects were seldom rendered correctly in real time. In computer games, simple texture blending or hardware fog was generally used to simulate light shafts. However, texture blending cannot handle the scenario when the viewer is totally inside the shaft volume. And hardware fog is totally wrong in terms of the physical model. Even though many physical models have been proposed to render light shafts, yet, few of them can be done in real time.

Recently, the processing speed of graphics card has been becoming faster and faster. In addition, the vertex processing unit (called vertex shader) and pixel processing unit (called pixel shader) have now become fully programmable. With the advent of programmability inside the graphics pipeline, a lot of expensive operations, which are used be done on CPU sequentially for each vertex, can now be carried out in parallel on GPU. Therefore, study of GPU-accelerated rendering is one of the most important research areas for real-time rendering.

In this paper, we proposed a rendering method of light shafts with atmosphere scattering by making use of GPU. The proposed method utilizes the programmability and parallel architecture of the GPU to display light shafts in real-time. Also, this method can handle shadow in atmosphere.

The paper is organized as follows. Previous work on rendering light shafts is discussed in Section 2 and an overview of the atmospheric scattering model and shading model we utilize is described in Section 3. In section 4, the core of our GPU-accelerated rendering method is proposed. Results and several examples are presented in Section 5. Finally, we draw the conclusion and discuss our future work.

2 Previous Work

A simple method to simulate the scattering and absorption due to atmospheric particles is to attenuate colors of objects according to the distance from the viewpoint. This method is computationally inexpensive and is implemented as one of the standard graphics APIs in OpenGL. However, this method is based on a heuristic function and is totally wrong in terms of actual physical phenomena. Hence, a more accurate model is required to simulate the atmospheric scattering effect.

To simulate the actual phenomena of atmospheric scattering, several models for atmospheric scattering have been proposed [1][2][3]. Based on Nishita's model, Dobashi proposed hardware-accelerated methods to render light shafts [4][5]. His method utilizes hardware texture blending and hardware Gouraud shading function to accelerate the rendering process and achieve interactive frame rate. However, due to the lack of programmability in the graphics pipeline, his method requires multiple passes and exploits little parallelism. Besides, in his method, the objects in the scene are required to render multiple times each frame to create the atmospheric shadow.

Our work uses the same shading model with Dobashi's, and is an extension and optimization on current programmable GPU. Also our proposed method only require two passes each frame and complex models in the scene are only rendered once to create the atmospheric shadow.

3 Overview of the Shading Model

Our method utilized a model proposed by Nishita [1]. We first describe the physical model briefly [6]. For the sake of simplicity, we only consider the case when there is only one light source. And our method can handle multiple light sources in a straightforward way.

Fig.1 shows the concept of atmospheric scattering. Here, a point light source is assumed. For parallel light source, the case will be simpler. In general, the intensity along a ray from the object reaching to the viewpoint is expressed by the Eq.1:

$$I_{eye}(\lambda) = I_{obj}(\lambda) * \beta_{\lambda}(T) + \int_{0}^{T} F_{\lambda}(\alpha) H(t) I_{p}(\lambda, t) \beta_{\lambda}(t) dt \tag{1}$$

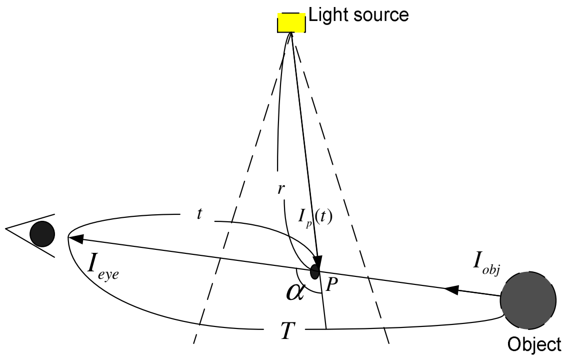


Fig. 1. Shading model for atmospheric scattering

where $I_{eye}(\lambda)$ is the intensity reaching the viewpoint, $I_{obj}(\lambda)$ is the intensity of an object, $\beta_\lambda(t)$ the attenuation ratio due to atmospheric particles between the viewpoint and a point P on the viewing ray, *t* the distance between the viewpoint and point P, *T* the distance between the viewpoint and the object, and $I_p(\lambda,t)$ the intensity of light from the light source reaching point P. $H(t)$ is a visibility function that returns the value 1 if the light source is visible from point P, or 0 otherwise. $F_\lambda(\alpha)$ is a phase function of the atmospheric particles and α is the phase angle (see Fig. 1). Because we only consider atmospheric scattering near to the ground, $\beta_\lambda(t)$ can be given by the Eq.2:

$$\beta_\lambda(t) = e^{-\beta_{sc}^\lambda t} \tag{2}$$

where β_{sc}^λ is the scattering coefficient for light with wavelength λ. And for point light source, $I_p(\lambda,t)$ is given by the Eq.3:

$$I_p(\lambda,t) = I_\lambda(\theta,\varphi)e^{-\beta_{sc}^\lambda r} / r^2 \tag{3}$$

where $I_\lambda(\theta,\varphi)$ is the intensity of light emanating from the light source toward the direction of point P and (θ,φ) indicates the direction. If the light source is a parallel light source, $I_p(\lambda,t) = I_0(\lambda)$, where $I_0(\lambda)$ is a constant. The phase function can be given by a weight sum of the Mie scattering and Rayleigh scattering, and the weight is selected according to atmosphere condition [7][8][9].

In Equation 1, the first term account for out-scattering, which is called aerial perspective, while the second term I_s account for in-scattering. The calculation of the first term is somewhat obvious, and can be easily implemented in shader. The calculation of the second term can be written as:

$$I_s = \sum F_\lambda(\alpha)H(t)I_\lambda(\theta,\varphi)\xi(\lambda,t) \tag{4}$$

$$\Delta I_s = F_\lambda(\alpha)H(t)I_\lambda(\theta,\varphi)\xi(\lambda,t) \tag{5}$$

$$\xi(\lambda,t) = e^{-\beta_{sc}^\lambda t}e^{-\beta_{sc}^\lambda r}\Delta t / r^2 \tag{6}$$

A fast method to calculate I_s is proposed in the following section.

4 GPU-Accelerated Rendering of Light Shafts

In order to compute the scattering effect of light shaft volume, virtual planes should be placed in front of the viewpoint firstly for sampling. Each virtual plane is parallel to the screen and is represented by a $n_u \times n_v$ lattices mesh (see Fig.2). For a viewing ray v, I_s is computed numerically by taking samples at intersections between the ray and the virtual planes. For point light source, $I_\lambda(\theta,\varphi)$ can be precompiled and used as a texture in rendering. Also we pre-calculate $F_\lambda(\alpha)$ and the result is then loaded into the graphic memory as a texture. Δt is a constant on the viewing ray since the sampling planes are placed at the same intervals.

To take into account shadow, $H(t)$ must be calculated for each lattice point. In our method, the camera is first placed at the position of the light source, then the scene is rendered and the color buffer is written with each pixel's depth value instead of each pixel's color. This color buffer, which contains the depth information of the objects in light space, is then used as a floating point depth texture to calculate $H(t)$ in later stage. The calculation of depth value can be done in shader. First, we calculate the position of each vertex in light space in vertex shader, and the result is assigned to a varying variable. Then, the fragment shader read the varying variable from output of the vertex shader, now this variable contains the position of the fragment in light space, and we write the z coordinate of this fragment into the color buffer. Also in this stage, since only the geometry information is required, texture and lighting are disabled to accelerate rendering.

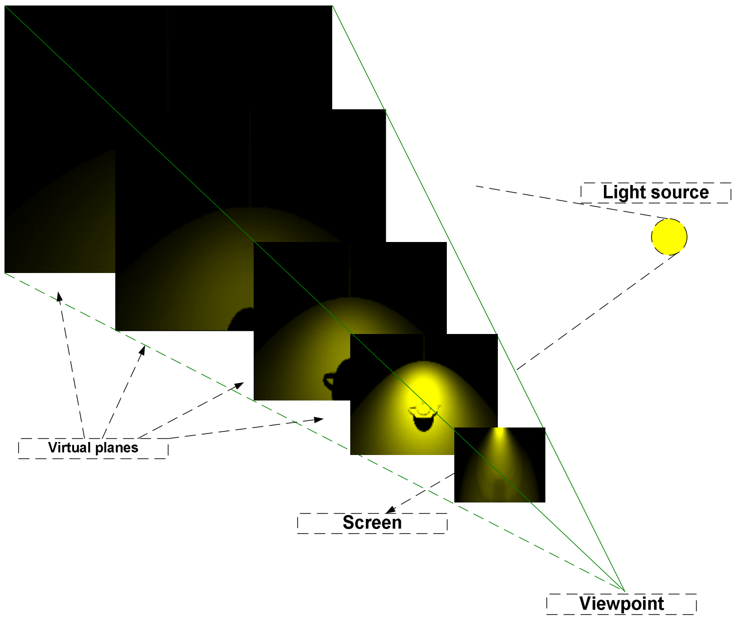


Fig. 2. Virtual planes and light computation on each lattices

In addition, every virtual plane is divided evenly into a $n_u \times n_v$ lattices mesh and $\xi(\lambda,\mathrm{t})$ is calculated for each lattice point. In our method, the calculation of the light intensity of each lattice point can be achieved in parallel by using a generic mesh refinement pattern [10]. This pattern is used to create additional inner vertices for each virtual plane by using the coordinates of its four corners. It is defined as a set of 2-tuple (u_t,v_t).

$$u_t = ((V_1 - V_0) \bullet (P - V_0)) * (P - V_0) /(|V_1 - V_0|^2 * |P - V_0|) \tag{7}$$

$$v_t = ((V_3 - V_0) \bullet (P - V_0)) * (P - V_0) /(|V_3 - V_0|^2 * |P - V_0|) \tag{8}$$

where $P(x_t,y_t,z_t)$ is a lattice point on a virtual plane, $V_{0,} V_1,V_2,V_3$ are the four corners of the virtual plane respectively (see Fig.3). Since each virtual plane is evenly divided into a $n_u \times n_v$ lattices mesh, the pattern is the same for all the virtual planes. Therefore, this pattern is uploaded into the graphics memory once during initialization, and is transferred from graphics memory to the vertex processing unit each time we draw a virtual plane. To calculate the world position and light intensity of each lattice point on the virtual plane, we send the world coordinates of V_0, V_1,V_2 and V_3 into the graphics pipeline as uniform variables and draw the pattern (each 2-tuple (u_t,v_t) in the pattern is regarded as a 2D vertex coordinate). In vertex shader, we read $V_{0,} V_1,V_2$ and V_3 and calculate the world coordinate of each lattice point using the Eq.9.

$$P = v_t * (V_3 - V_0) + u_t * (V_1 - V\ ) + V_0 \tag{9}$$

$\xi(\lambda,\mathrm{t})$ is then calculated using the world coordinate. Finally, $\xi(\lambda,\mathrm{t})$ of other points on the virtual plane can be interpolated by using their adjacent lattice points, and the interpolation is performed using Gouraud shading.

In the fragment shader, we compute the depth value (in light space) of the fragment and compare it with stored value in the depth texture (the lookup of the stored value can be achieved using projective texture mapping algorithm), if the fragment's depth in light space is greater than the stored value, it is then in shadow, $H(t)=0$, otherwise, $H(t)=1$. At last, $F_\lambda(\alpha)$ and $I_\lambda(\theta,\varphi)$ are read back from textures respectively and used to calculate ΔI_s together with $H(t)$ and $\xi(\lambda,\mathrm{t})$.

The procedure for our rendering method using OpenGL is summarized as follows:

1) Preprocess $F_\lambda(\alpha)$ and $I_\lambda(\theta,\varphi)$ as textures and upload them into the graphics memory. Precompute the pattern of the lattice mesh and upload it into the graphics memory through a static Vertex Buffer Object.
2) For each frame, attach a floating point depth texture to a frame buffer object, and place the camera at the position of the light source, then the scene is rendered into the frame buffer object and color buffer is updated with the depth value of the objects. Since we only care about the depth value, texture and lighting can be disabled to accelerate rendering.
3) For each virtual plane, send the coordinates of its four corners to vertex shader as uniform variables, and draw the pattern of the lattice mesh. In the shader, the process of each lattice point is as the following:
   a) Calculate the eye coordinates of the lattice points using the four corners of the virtual plane and the refinement pattern.

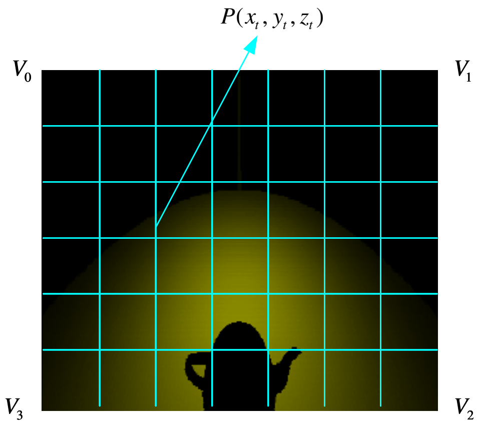


Fig. 3. Virtual plane refinement pattern

b) Calculate the light coordinate of the lattice point. By using the projective texture mapping technique, we can lookup the corresponding light intensity I_λ and depth value **z** (in light space) for each lattice point in texture $I_\lambda(\theta,\varphi)$ and the depth texture respectively.

c) Calculate the depth of the lattice point in light coordinate, and compare it with **z**. If **z** is smaller, this lattice point is in shadow. Otherwise, we lookup $F_\lambda(\alpha)$ from texture and compute ΔI_s.

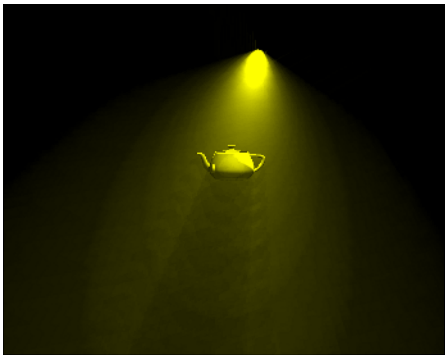
4) All the virtual planes are rendered using additive blending function.

5 Experiments

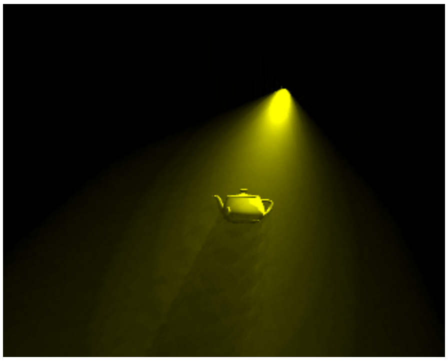
Our experiments are performed on a PC workstation (Pentium 4 3.2GHz HT, 1 Gigabyte RAM) with ATI X800 GTO. All the images are rendered at the resolution of 800x640. Some results are shown in table 1. We find out that the efficiency of our method is inversely proportional to the number of virtual planes. Also, as the refinement pattern becomes finer, the computational time increases. This should be due to

Table 1. Statistics data of light shaft rendering

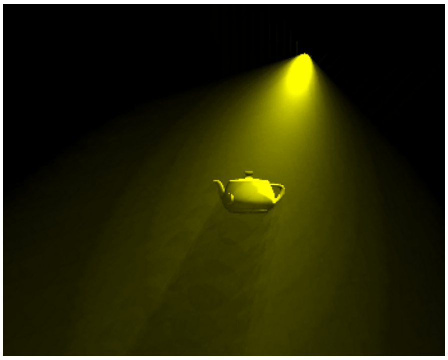
Figure No.	Virtual planes	Lattices points	Depth texture	Light-map	Frame rate (fps)
4(a)	50	32x32	256x256	64x64	30
4(b)	50	64x64	256x256	64x64	26
4(c)	50	64x64	512x512	64x64	24
4(d)	50	64x64	512x512	256x256	22
4(e)	100	32x32	256x256	64x64	16
4(f)	100	96x96	512x512	128x128	10
5(a-d)	100	96x96	512x512	128x128	9



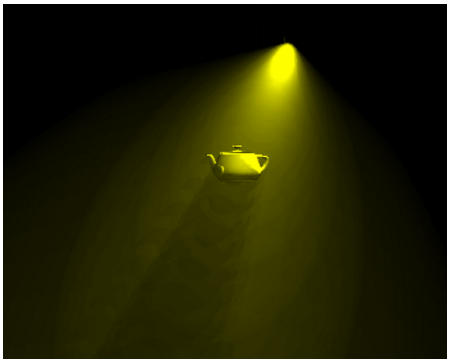
(a)



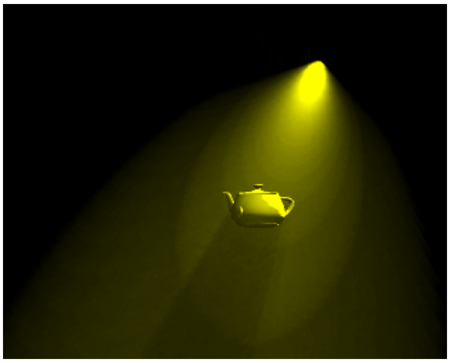
(b)



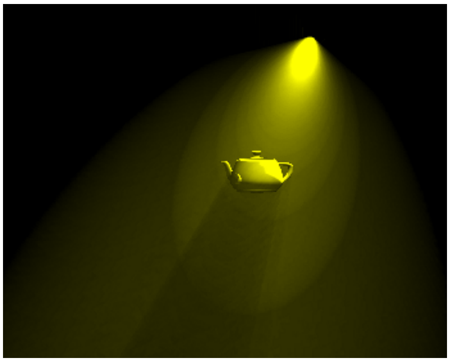
(c)



(d)



(e)



(f)

Fig. 4. Rendering of the light shaft using different rendering setting listed in Tab. 1

a bottleneck between the vertex fetch unit and the vertex processing unit, since there is usually 6-8 parallel vertex processing pipeline while the vertex fetch unit might fetch more vertices at a time. Besides, as we increase the resolution of the depth

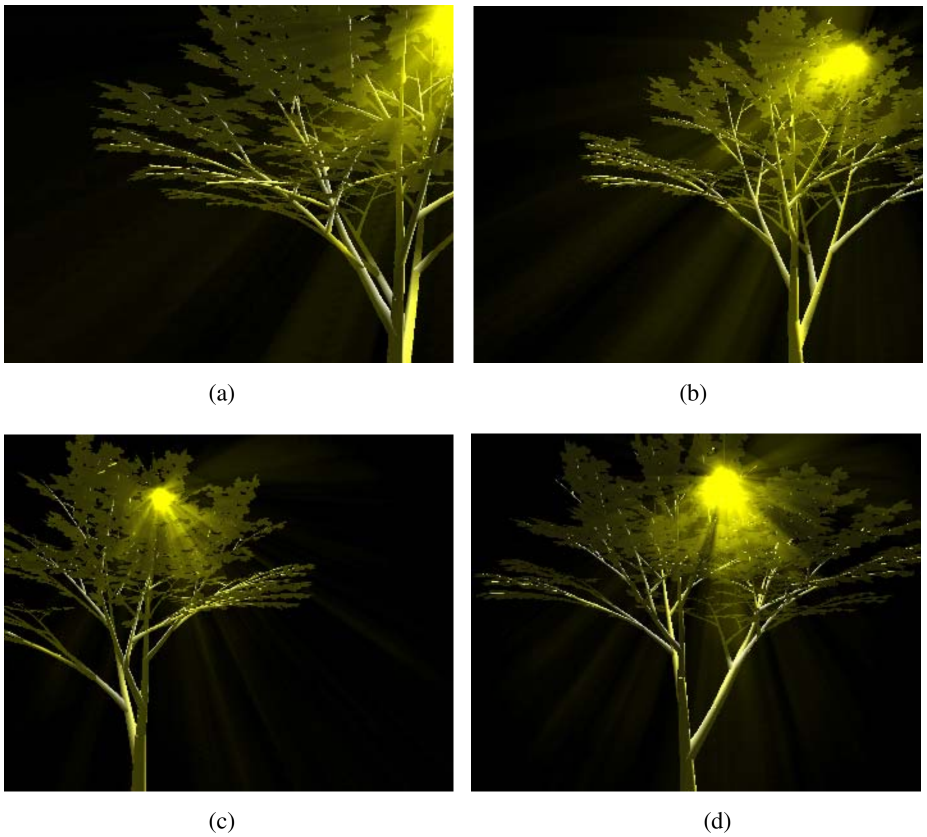


Fig. 5. Rendering of the light shaft when viewpoint located in the shaft volume under a complex scene

texture, the performance drops a little bit. This should be due to the bandwidth bottleneck between GPU and graphics memory. What is more, in Fig. 5(a-d), the scene complexity is around 43k triangles, and the performance drops only 10% when compared with the scene showed in Fig. 4.

6 Conclusion

In this paper, we have proposed a new method of rendering light shafts with atmospheric scattering in real-time. The proposed method makes use of the programmability and parallel architecture on modern GPU to achieve fast frame rate and photorealistic effect. The advantages of our method are as follows.

1) By utilizing the depth texture, our method requires two passes for each frame. The depth texture is only rendered at the beginning of each frame, and is read multiple times for each virtual plane. Hence, the objects in the scene casting

shadows are rendered twice in total at each frame. (One is for the depth texture, the other is for the scene displayed on the screen)

2) Our method utilizes a mesh refinement pattern, and this pattern is the same across all the virtual planes. It is uploaded as a static vertex buffer into graphics memory and reused multiple times when we draw a virtual plane. Therefore, the footprint between CPU and GPU is greatly reduced, making full use of the fast transfer rate between GPU and graphics memory (around 35GB/s).
3) In the past, $\xi(\lambda, \mathrm{t})$ of each lattice point is calculated sequentially on CPU. In our method, by making use of a mesh refinement pattern, $\xi(\lambda, \mathrm{t})$ of lattice points can be calculated in parallel inside the graphic pipeline.

In future work, we need to develop a method to render multiple scattering in real-time [9]. Also, the sampling error due to the numerical integral in Equation 4 needs to be solved in order to create more photorealistic images.

Acknowledgements

This research work is supported by National Key Project of Fundamental Research (973 Project No. 2004CB719403), NSFC (60573151, 60473100).

References

1. T. Nishita, Y. Miyawaki, E. Nakamae, A Shading Model for Atmospheric Scattering Considering Distribution of Light Sources, Computer Graphics, 21(4):303–310, 1987.
2. R. V. Klassen. Modeling the Effect of the Atmosphere on Light. ACM Transactions on Graphics, 6(3): 215-237, July 1987.
3. K. Kaneda, T. Okamoto, E. Nakame and T. Nishita. Photorealistic image synthesis for outdoor scenery under various atmospheric conditions. The Visual Computer 7, 5 and 6, 247-258, 1991.
4. Y. Dobashi, T. Yamamoto, T. Nishita. Interactive rendering method for displaying shafts of light, In Pacific Graphics, 2000.
5. Y. Dobashi, T. Yamamoto, T. Nishita. Interactive rendering of atmospheric scattering effects using graphics hardware. Proceedings of the conference on Graphics hardware, 2002.
6. A. Preetham, P. Shirley, B. Smits, A Practical Analytic Model for Daylight, Proc. SIGGRAPH'99, 91–199, 1999.
7. L. Rayleigh. On the scattering of light by small particles. Philosophical Magazine 41, 447-451, 1871.
8. K.N. Liou. An Introduction to Atmospheric Radiation. International Geophysics Series Volume 84. Academic press. 2002.
9. T. Nishita, Y. Dobashi, K. Kaneda, and H. Yamashita. Display method of the sky color taking into account multiple scattering. In Pacific Graphics, pp. 117-132, 1996.
10. T. Boubekeur, C. Schlick. Generic Mesh Refinement on GPU, in: Proceedings of ACM SIGGRAPH/Eurographics Graphics Hardware, 2005.

Learning the Stylistic Similarity Between Human Motions

Yu-Ren Chien and Jing-Sin Liu

Institute of Information Science
Academia Sinica, Taiwan
yrchien@ntu.edu.tw, liu@iis.sinica.edu.tw

Abstract. This paper presents a computational model of stylistic similarity between human motions that is statistically derived from a comprehensive collection of captured, stylistically similar motion pairs. In this model, a set of hypersurfaces learned by single-class SVM and kernel PCA characterize the region occupied by stylistically similar motion pairs in the space of all possible pairs. The proposed model is further applied to a system for adapting an existing clip of human motion to a new environment, where stylistic distortion is avoided by enforcing stylistic similarity of the synthesized motion to the existing motion. The effectiveness of the system has been verified by 18 distinct adaptations, which produced walking, jumping, and running motions that exhibit the intended styles as well as the intended contact configurations.

1 Introduction

Human motions come in various types or styles, such as walking, limping, running, jumping, etc. Animators are always going to great pains to ensure that the produced character motions exhibit styles that match the exact styles they intend. Even if motions matching animators' stylistic intentions occasionally exist in motion capture databases, such canned motions usually cannot be directly reused due to the inability of these very motions to (geometrically) fit in the current scenes. What underlies an animator's effort to check a produced style against an intended or canned style, is a perceptual similarity between two motions, which we call the *stylistic similarity* between human motions. This work aims to automate this similarity judgment made by animators—we believe that such automation would greatly facilitate character animation. Specifically, automated stylistic similarity judgments can be applied to *human motion adaptation*, where a canned or captured motion with the desired style, which we call the *example motion*, is adapted to the geometry of the current scene (see Figure 1) with stylistic details preserved. We present a framework for human motion adaptation where stylistic similarity of the synthesized motion to the example motion is ensured by building the automated similarity judgment into the spacetime constraints formulation [1].

In this work, we regard the similarity judgment as a classification task, where the observer decides whether two motions are stylistically similar or not. Instead of gathering training motion pairs of both classes and adopting a two-class pattern recognition technique like SVM, we utilize a comprehensive collection of stylistically similar motion pairs made available by motion capture, and take an unsupervised learning approach

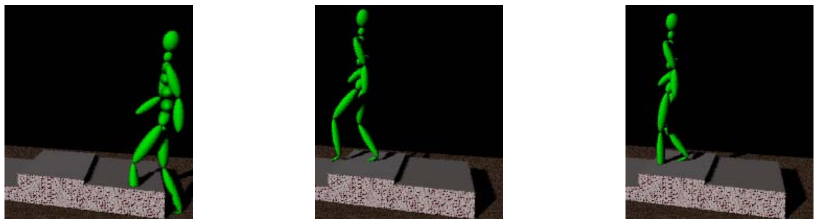


Fig. 1. A system for human motion adaptation adapts a user-supplied (example) motion to a new environment while preserving its stylistic details. **Left**: A typical example motion (walking on the flat ground, with the left foot raised off the ground in this snapshot), which is geometrically undesirable with respect to the intended environment shown here, in that the foot-contact configurations deviate from the targets on the marble steps so much that severe penetrations into the marble steps occur. **Middle**: The intended new motion for this particular scenario (example motion and intended environment), which has been adapted to the surface of the marble steps without being stylistically distorted. **Right**: A stylistically undesirable new motion for the same scenario, where the foot contacts have been adapted to the surface of the marble steps while the left foot penetrates into the marble steps due to the abnormally straight pose of the left leg.

to automating this similarity judgment, using single-class SVM (1-SVM) [2] and kernel PCA (KPCA) [3] to model the distribution of stylistically similar motion pairs. Moreover, in order to extract from two motions appropriate features for use with these statistical techniques, we propose a set of rules for segmentation of human motions and alignment of segmented motions, which rules form a structural paradigm for treating human motions that is novel in the world of character animation.

One concern about our model of stylistic similarity would be the lack of a formal definition of what we mean by "style," which term always derives its definition from some subjective taxonomy implied by its context. Since the algorithms we adopt for building the model, i.e. 1-SVM and KPCA, generalize entirely from the training data, the "styles" considered in this work are actually defined by the exact labeling system used in the CMU Graphics Lab Motion Capture Database. In practice, we collected walking, running, and jumping data exactly as they are labeled in the database. We believe that the proposed technique for modeling stylistic similarity can be applied to any specific setting of stylistic taxonomy, or even any other perceptual (perhaps not visual) modalities, as long as appropriate data and features can be provided.

2 Related Work

Extraction of stylistically relevant features from human motion is essential to the effectiveness and efficiency of the following two tasks:

- **Style-based posture estimation:** In [4], each transition pose in jumps was represented by some centers of body-part mass while estimated. In [5], joint angles, vertical components in pelvis orientation, and their time-derivatives were extracted for estimation of the probability density of human posture in a particular style of motion; and

- **Human activity recognition:** In [6], joint positions were extracted for short-time recognition of motion styles. In [7], a fixed number of frames were randomly sampled from the 2-D motion; from each of these frames, global 2-D angular poses/velocities were extracted. In [8], feature extraction from an arm motion was performed by sampling the 2-D hand trajectory at every impulse of the hand acceleration signal.

There has been much research into human motion adaptation. Certain details in the example motion can be preserved in the new motion by minimizing a weighted sum of squared changes in motion parameters [9,10,11,12], or by iteratively modifying the (initially example) motion with the gradient method [13] or with a non-statistical variant of the Kalman filter [14]. In [15], [16], and [17], force patterns were extracted from the example motion and enforced in the new motion.

3 Segmentation and Alignment

In this work, we decompose each motion into a set of *motion segments*, rather than a sequence of postures, as the basic units of processing. The rationale behind this choice is that motion segments are more psychologically relevant than postures—we do not perceive and control our motions as individual postures, but as such basic movements as steps and twists [18]. Also inspired by the fact that we would at some times observe the motions of various body parts as a whole, and at other times focus on the motion of a particular limb, we break each motion both into *whole-body motion segments* and into *limb-specific motion segments*.

Segmentation of the motion of a specific limb is based on the interaction of its end-effector with the environment. There are two states to this interaction: the in-contact state and the free state. The motion is segmented at every state transition, e.g. at the heel-down frame and the toe-off frame of each foot contact, so that the resulting motion segments sequentially alternate between the two states.

Segmentation of a whole-body motion is achieved by *fusing limb-specific motion segmentations*, which in turn consists in grouping by concurrence all the limb-specific motion segments (originally grouped by limb). Let the sets A and B denote the frame numbers spanned by two limb-specific motion segments, and assume that $|\mathsf{A}| \geq |\mathsf{B}|$ ($|\cdot|$ denotes the set cardinality). The two segments are declared to be in the same group if and only if $|\mathsf{A} \cap \mathsf{B}| > \frac{|\mathsf{B}|}{2}$, where $\cap$ denotes the set intersection. We define a whole-body motion segment for each of the resulting (mutually exclusive) groups by assigning the *union* of the limb-specific frame spans to the whole-body frame span.

Consider two motions $\mathscr{M}_1$ and $\mathscr{M}_2$ of the same limb that share the same state sequence of length N, i.e. that start with the same state of interaction with the environment and both undergo exactly $N-1$ state transitions. To relate $\mathscr{M}_1$ and $\mathscr{M}_2$ in a structural manner, we align them by the state sequence, giving N pairs of aligned motion segments. In general for two alignable motions, the i-th (whole-body or limb-specific) segment in one motion can only be aligned with the corresponding (i-th) segment in the other. A pair of aligned motion segments extracted from a motion capture database generally have different durations.

4 A Framework for Human Motion Adaptation

4.1 Parameterization

The new motion to be optimized is represented by a vector made up of the following variables:

- 6 rigid DOFs per frame for pelvis motion; and
- 60 standardized *principal components* for each motion segment of each limb with which the character touches the environment. Any limb or torso that never touches the environment inherits its motion from the example motion without taking up any variables.

A concatenation of a 25-sample version of the angular signals represented by the 60 components can be computed by transforming the components according to a learned subspace [18].

For subspace learning, we have collected 98 motion clips of such types as climbing, jumping, running, walking, etc. By collecting limb-specific motion segments across all the clips, we have as training data 514 (507, resp.) segments for the left leg (right leg, resp.). After performing PCA on the training patterns extracted from the training segments [18], we found the first 60 of the 175 principal components for each (7-DOF) leg to account for about 99.9% of the total variation.

The low dimensionality of this motion representation is intended to avoid any prohibitive amount of computation time or failure of convergence in the optimization, which would generally be incurred by simply representing the motion by tens of postural variables per frame.

4.2 Constraints and Objectives

The synthesized motion $\mathbf{x}^*$ minimizes the function

$$F(\mathbf{x}) = w_p \cdot P(\mathbf{x}) + w_c \cdot C(\mathbf{x}) + w_q \cdot Q(\mathbf{x})$$

subject to the constraints $\mathbf{S}(\mathbf{x}) \geq \mathbf{0}$ and $\mathbf{b}_l \leq \mathbf{V}(\mathbf{x}) \leq \mathbf{b}_u$.

The objective function $P(\cdot)$ represents the *stylistic similarity* of the new motion to the example motion, giving the sum-square value of $N_l + N_w$ nonnegative distortion scores based on the KPCA, where N_l is the number of limb-specific motion segments represented by $\mathbf{x}$, and N_w is the number of whole-body motion segments in each of the new motion and the example motion. Each distortion score measures the stylistic distortion of a motion segment in the new motion from its aligned counterpart in the example motion, so that the score is close to zero if and only if the aligned pair is stylistically similar. In addition, $C(\cdot)$ represents the sum-square distance of end-effectors from user-specified targets during contacts, and $Q(\cdot)$ penalizes 1) excessive contact torques about centers of support [4,19], and 2) loss of balance [4] at frames specified by the user for balancing.

The constraint function $\mathbf{S}(\cdot)$ also represents the *stylistic similarity* of the new motion to the example motion, giving a vector of $N_l + N_w$ similarity scores based on the 1-SVM. Each similarity score measures how a pair of aligned motion segments (a segment in the new motion and its aligned counterpart in the example motion) are stylistically similar to each other, so that the score is nonnegative if and only if the pair is stylistically similar. Moreover, $\mathbf{V}(\cdot)$ is intended to realize joint limits, intersegmental continuity, smoothness of pelvis motion, and flight-phase dynamics.

Since $F(\cdot)$ has a sum-square structure, this particular nonlinear program can be treated as a constrained nonlinear least squares problem.

5 Stylistic Similarity

In this section, we develop algorithms for constructing from motion capture data computational models of the stylistic similarity between motion segments. These algorithms are based on the abstraction of a space of all possible motion segment pairs, and the characterization of a dichotomy in that space that represents the difference between stylistic similarity and stylistic distortion. To be specific, we take advantage of the abundance of instances of stylistic similarity in a motion capture database to characterize the region occupied by all possible stylistically similar pairs, which we call the *similarity region*. Our intended models of stylistic similarity will be given by approximations of the similarity region.

To approximate the geometry of a region in the 3-D space, one might sample the region and simply fit a surface to the boundary samples; however, if the region is part of any surface, it would be more effective to also fit one or two surfaces to all the samples. In view of this, we apply two approximation techniques to the similarity region: the 1-SVM [2] and the KPCA [3], which respectively solve the above two classes of approximation problems, giving a *supporting hypersurface* and a set of *fitted hypersurfaces* for the sampled similarity region.

5.1 Feature Extraction

In this section, we describe the actual features extracted to represent each point in the space of all possible motion segment pairs. Specifically, we consider multiple spaces of motion segment pairs: one for pairs formed by whole-body motion segments, and the others for pairs formed by limb-specific motion segments (one space per limb). The features extracted for each space will underlie the approximation of the corresponding similarity region.

Whole-Body Features. The procedure defined in this section takes a pair of whole-body motion segments (A, B) as input, and gives as output a feature vector that encodes the Cartesian postural evolutions of major body parts in A and B. As shown in Figure 2, the extraction process starts by linearly interpolating each of A and B at K uniformly spaced instants to give K postures (totally $2K$ postures for A and B). For each such posture, consider a reference frame that has origin at pelvis, y-axis upward, and z-axis anterior (see Figure 2), which serves to remove all the global information, except the balance information, from the features for the posture. The process then computes (using the mass distribution data reported in [20]) centers of upper-body, left-leg, and right-leg mass (see Figure 2) with respect to the reference frame for each of the $2K$ postures. Concatenating the COM position vectors for all the three body parts and for

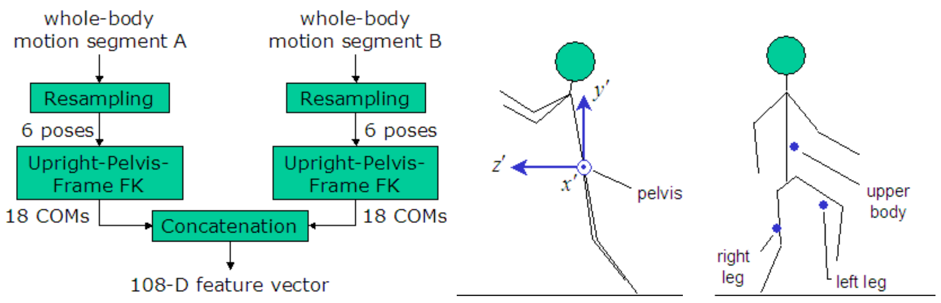


Fig. 2. Whole-body feature extraction. **Left:** The procedures. (FK stands for forward kinematics.) **Middle:** The reference frame. The character exactly faces the left. **Right:** The COMs computed.

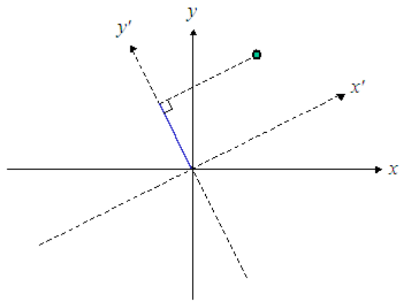
all the postures, the process finally produces an $18K$-dimensional vector as the feature vector. We set K to 6 in all experiments.

Limb-Specific Features. The feature vector for a pair of motion segments (A, B) of a limb is 30-dimensional, composed of the first 15 standardized principal components for each of A and B. The 15 components are extracted from each segment by the analysis procedures given in Section 4.1, except with a further truncated subspace.

5.2 Building the Models

To sample the similarity region in the space defined by each extracted set of features, i.e. to gather training patterns, we have collected about 70 pairs of captured motion clips, each pair formed by two motions *of the same type* (walking, jumping, or running) and with the same sequence of end-effectors in contact with the environment (e.g. two walks both starting with a left-foot contact and lasting for five steps, or two jumps both composed of a doubly supported takeoff, a flight stage, and a doubly supported landing, so that their segmentations can be aligned). To extract three sets of training patterns from the clips (one dataset for whole-body features, one for left-leg features, and the other for right-leg features; in each dataset we have approximately 90 patterns from walking, 70 from jumping, and 90 from running), we applied the whole-body feature extraction procedure described in Section 5.1 to each pair of aligned whole-body motion segments found in each pair of clips, and the limb-specific procedure in Section 5.1 to each pair of aligned limb-specific (leg) motion segments found in each pair of clips.

Models Based on the 1-SVM. As the first part of our modeling efforts, for each dataset $\{\mathbf{p}_i \in R^n : i = 1, \ldots, I\}$, we fit a hypersurface $\{\mathbf{p} \in R^n : f_s(\mathbf{p}) = 0\}$ to the boundary samples in the dataset such that $f_s(\mathbf{p}_i) \geq 0$, $\forall i \in \{1, \ldots, I\}$, i.e. all the training patterns lie on the same side of the *supporting hypersurface*, using the *single-class* SVM training routine in the OSU Support Vector Machines Toolbox, which is based on the LIBSVM library. 1-SVM [2] and the better-known two-class SVM differs in that instead of finding a separating hypersurface between the respective supports of two groups of training patterns, the 1-SVM algorithm simply estimates the support of a single group of training patterns, i.e. its purpose is to detect novelty given a set of usual patterns.



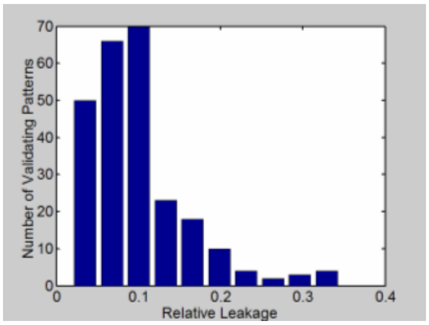


Fig. 3. Left: Illustrating the feature-space distance of a testing pattern from the subspace spanned by the lower-order principal axes. The x-y plane represents the ∞-D feature space. The x'- and y'-coordinates represent the lower-order and the higher-order principal components, respectively. The dot represents the testing pattern. The blue line segment on the y'-axis represents the distance. **Right**: Cross-validation result for the hypersurfaces fitted to the whole-body dataset.

In the theory of 1-SVM, the supporting hypersurface can be represented by a supporting hyperplane in an infinite-dimensional space related to the original space by a radial basis function kernel, $f_k(\mathbf{x},\mathbf{y}) = \exp\left(-\gamma\cdot\|\mathbf{x}-\mathbf{y}\|^2\right)$. Here we adopt this kernel function with the parameter γ tuned to give the best cross-validation accuracy possible. To estimate the modeling accuracy of each supporting hypersurface learned by the 1-SVM algorithm, we have conducted 20-fold cross-validation on the corresponding dataset. The probability of leakage of sample $\mathbf{p}\in R^n$ beyond the whole-body (left-leg, right-leg, resp.) supporting hypersurface, $\text{Prob}\left[f_s(\mathbf{p})<0\right]$, is estimated to be 0.048 (0.0332, 0.0413, resp.) with the 95% confidence interval calculated to be $[0.025, 0.0823]$ ($[0.0144, 0.0644]$, $[0.02, 0.0747]$, resp.).

Models Based on the KPCA. As the second part of our modeling efforts, we fit a set of hypersurfaces to the samples in each dataset by means of the KPCA algorithm [3]. In the theory of KPCA, one can perform PCA on the training patterns in the same ∞-D feature space as defined for 1-SVM modeling above. For each resulting principal component that has small variance, we have a corresponding hyperplane that is fitted to the training patterns in the ∞-D space: $\{\mathbf{p}\in R^\infty:\ \mathbf{e}\cdot(\mathbf{p}-\boldsymbol{\mu})=0\}$, which is perpendicular to the corresponding principal axis $\mathbf{e}\in R^\infty$, passes through the mean $\boldsymbol{\mu}\in R^\infty$, and defines a corresponding input-space *fitted hypersurface*. One may find principal components to have small variances starting from order O_c. For all these small components, which we call *higher-order principal components*, we have a corresponding set of input-space hypersurfaces that are all fitted to the training patterns, so that each training pattern is known to lie on the intersection of the set of hypersurfaces. Consider a testing pattern represented in the ∞-D space by the principal components $\mathbf{z}=(z_1,z_2,\ldots)$. We measure how close the pattern is to the intersection of input-space fitted hypersurfaces by the proportion of its feature-space squared norm contributed by the higher-order principal components $(\|\mathbf{z}\|^2-\sum_{i=1}^{O_c-1} z_i^2)/\|\mathbf{z}\|^2$, which we call the *relative leakage* of the pattern with respect to the fitted hypersurfaces. As depicted in Figure 3, the numerator can be interpreted as the squared distance of the pattern from the subspace spanned by the lower-order principal axes. Note that as an inner product in the centered ∞-D space, $||\mathbf{z}||^2$ can be computed via $f_k(\cdot,\cdot)$ and centering [3]. In our experiments, the cut-off order O_c is set to 16 for the whole-body dataset and to 22 for each single-leg dataset.

To estimate the modeling accuracy of each set of fitted hypersurfaces learned by the KPCA algorithm, we have conducted 20-fold cross-validation on the corresponding dataset. For the hypersurfaces fitted to the whole-body dataset, we plot the histogram of the relative leakage over all validating patterns in Figure 3; for the hypersurfaces fitted to the single-leg datasets, the same histograms are again computed, as not shown here due to the space limit. For every set of fitted hypersurfaces, the average relative leakage over all the validating patterns is found to be roughly 0.1, and about 90% of the patterns are found to have relative leakages below 0.2.

6 Adaptation Experiments

In our implementation of the adaptation system presented in Section 4, for better efficiency of the optimization, we apply the transformation proposed by Schittkowski [21] to the constrained nonlinear least squares problem defined in Section 4.2. The resulting nonlinear program is coded in ANSI Fortran 77, augmented with sparse derivative computation by the ADIFOR 2.0 automatic differentiation tool, and solved by the sparse nonlinear programming routine in the NAG Fortran Library, which is based on the SNOPT package described in [22], on a Pentium-4 1.8-GHz PC running Red Hat (7.2) Linux with 512-MB RAM. Note that low-dimensional representation of human motion serves here to prevent variables from significantly outnumbering constraints, thereby favoring the efficiency of SNOPT and ADIFOR-generated codes. In our experiments, every adaptation took less than thirty minutes.

To evaluate the effectiveness of our adaptation system, we have tested it on 18 distinct motion adaptation tasks. In these tasks, the example motions are of the types walking, jumping, and running (6 tasks for each type) and apart from those used in preparation of the datasets from which the models of stylistic similarity were learned, and most of them are each composed of 3 or 4 whole-body motion segments. We specified contact targets such that each example motion significantly deviates from the associated targets in the new environment, and the targets still look attainable for the specific activity in the example motion, thereby controlling the task difficulty. By playing back and watching each of the 18 synthesized motions, we found that each motion exhibits the intended style, as well as a realistic contact during each contact event. All these tasks were performed with the same parameter setting, except that in running tasks only 30 standardized principal components were used to represent each leg motion segment in order for the number of variables to roughly match the relatively small number of constraints in running tasks. Only ending frames in jumps were marked for balancing.

To verify the necessity of enforcing stylistic similarity to the example motion, we created a crippled version of our system by dropping from the nonlinear program the functions representing the similarity, i.e. $\mathbf{S}(\cdot)$ and $P(\cdot)$, and repeated all the above 18 tasks using this crippled version instead. The new motions resulting from this verification were each found to be stylistically distorted. Manifestations of such distortion include abnormal poses, abnormal variations in speed of movement, and abnormal directions of movement. On the other hand, robustness in attaining geometrical goals was also impaired—two of these motions were found to be geometrically highly undesirable. One interesting implication of this verification is that collisions and kinematical singularities are usually accompanied by stylistic distortion and can be adequately avoided by enforcing stylistic similarity to the example motion, which is obvious when we notice that abnormal poses brought about penetrations in several tasks, and that local minimums of the geometrical sum-square error are a specific form of kinematical singularities.

7 Discussion

We have presented a motion adaptation system that is sensitive to stylistic fidelity in its quest for the optimal solution. In this final part of the paper, we discuss the uniqueness of this system by providing some in-depth comparisons with the system proposed in [12]:

- Both muscle forces and contact forces are bound-constrained in [12]. We also constrain contact forces in our system. However, without computing muscle forces, we let the new motion imitate the example motion according to the data-driven model of stylistic similarity, which we believe has the effect of capturing not only the bounds on, but also the patterns of, muscle forces. Such pattern-following mechanism is also present in [12] in the form of minimum sum-square difference between Cartesian representations of the example motion and the new motion.
- Both systems generate new motions by solving spacetime optimization with the same sparse nonlinear programming solver, and, for better convergence of the optimization, simplify the underlying postural parameterization according to the specific behavioral characteristics in the example motion. We also build into our parameterization a data-driven set of constraints that account for the inherent spacetime characteristics of all human motions, thereby further lowering the dimensionality.
- Both systems are shown in extensive experiments to accurately adapt jumping/running examples to new environments while preserving stylistic quality. We additionally present style-preserving adaptations for walking motions.

Acknowledgments

The data used in this project was obtained from `mocap.cs.cmu.edu`. The database was created with funding from NSF EIA-0196217. The ADIFOR 2.0 System was obtained from Rice University and the University of Chicago as Operator of Argonne National Laboratory. This work has been supported by NSC 93-2212-E-001-001, NSC 92-2212-E-001-001, and NSC 91-2212-E-001-001.

References

1. Witkin, A., Kass, M.: Spacetime constraints. In: SIGGRAPH. (1988) 159–168
2. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Computation **13** (2001) 1443–1471

3. Schölkopf, B., Smola, A., Müller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation **10** (1998) 1299–1319
4. Liu, C.K., Popović, Z.: Synthesis of complex dynamic character motion from simple animations. In: SIGGRAPH. (2002) 408–416
5. Grochow, K., Martin, S.L., Hertzmann, A., Popović, Z.: Style-based inverse kinematics. ACM Trans. Graph. **23** (2004) 522–531
6. Arikan, O., Forsyth, D.A., O'Brien, J.F.: Motion synthesis from annotations. ACM Transactions on Graphics (TOG) **22** (2003) 402–408
7. Ben-Arie, J., Wang, Z., Pandit, P., Rajaram, S.: Human activity recognition using multidimensional indexing. IEEE Trans. Pat. Anal. Mach. Intel. **24** (2002) 1091–1104
8. Rao, C., Yilmaz, A., Shah, M.: View-invariant representation and recognition of actions. International Journal of Computer Vision **50** (2002) 203–226
9. Gleicher, M.: Motion editing with spacetime constraints. In: Proceedings of the 1997 symposium on Interactive 3D graphics. (1997) 139–ff.
10. Gleicher, M.: Retargetting motion to new characters. In: SIGGRAPH. (1998) 33–42
11. Lee, J., Shin, S.Y.: A hierarchical approach to interactive motion editing for human-like figures. In: SIGGRAPH. (1999) 39–48
12. Popović, Z., Witkin, A.: Physically based motion transformation. In: SIGGRAPH. (1999)
13. Sulejmanpašić, A., Popović, J.: Adaptation of performed ballistic motion. ACM Trans. Graph. **24** (2005) 165–179
14. Tak, S., Ko, H.S.: A physically-based motion retargeting filter. ACM Trans. Graph. **24** (2005)
15. Pollard, N.S., Behmaram-Mosavat, F.: Force-based motion editing for locomotion tasks. In: Proceedings of the IEEE International Conference on Robotics and Automation. (2000)
16. Pollard, N.S.: Simple machines for scaling human motion. In: Computer Animation and Simulation '99. (1999)
17. Liu, C.K., Hertzmann, A., Popović, Z.: Learning physics-based motion style with nonlinear inverse optimization. ACM Trans. Graph. **24** (2005) 1071–1081
18. Fod, A., Matarić, M.J., Jenkins, O.C.: Automated derivation of primitives for movement classification. Autonomous Robots **12** (2002) 39–54
19. Fang, A.C., Pollard, N.S.: Efficient synthesis of physically valid human motion. ACM Transactions on Graphics (TOG) **22** (2003) 417–426
20. de Leva, P.: Adjustments to Zatsiorsky-Seluyanov's segment inertia parameters. J. Biomechanics **29** (1996) 1223–1230
21. Schittkowski, K.: Solving constrained nonlinear least squares problems by a general purpose SQP-method. In Hoffmann, K.H., et al., eds.: Trends in Math. Optim. (1988) 295–309
22. Gill, P.E., Murray, W., Saunders, M.A.: SNOPT: An SQP algorithm for large-scale constrained optimization. Technical Report NA–97–2, Dept. Mathematics, UCSD (1997)

Effects of Layer Partitioning in Collaborative 3D Visualizations

Lars Winkler Pettersson[1], Andreas Kjellin[2], Mats Lind[2], and Stefan Seipel[1,3]

[1] Dept. of Information Technology, Uppsala University
lwp@it.uu.se
[2] Dept. of Information Science, Uppsala University
{mats.lind, andreas.kjellin}@dis.uu.se
[3] Dept. of Mathematics, Natural and Computer Sciences, University of Gävle
ssl@hig.se

Abstract. Display technologies that support multiple independent views of the same co-located 3D visualization volume make new forms of collaboration possible. In this field of research, until now most efforts have focused on technical solutions and their applications. The main contribution of this paper is the results from a study comparing integral and partitioned 3D content in a head coupled stereoscopic environment through independent views of a shared 3D visualization.

In our study we used a geospatial task that was solved by ten pairs of collaborating individuals (dyads). We measured task performance by time and error rate for the dyads in two main conditions: a) an integral visualization that presented a map in the display surface and four layers at different depths below the display surface to each of the observers, and b) a partitioned visualization, where two mutually exclusive subsets of the layers were presented to each of the observers together with the map in the display surface.

The results from the study showed significant differences in regard to performance times between the two conditions. Task performance was significantly better in the condition with layer partitioning. Partitioned visualizations can thus, at least in some cases, improve performance in tasks requiring collaboration between users.

1 Introduction

Collaboration between two or more people in the same location can, in addition to the use of direct communication such as spoken language or body language, be supported by artifacts in our environment. Artifacts can help mediate information in terms of their position, form or other qualities. In the history of military operations, artifacts have been used in mission planning to describe previous, current or future situations. Artifacts have for example been rocks, sand, paper and pencils, symbols, maps, etc. Today we rely on the use of computer systems as artifacts to process and present information. For instance, the paradigm of network centric warfare requires computer systems to process and provide a wealth of information even for small geographic areas. Other fields that rely on computer systems to gather and present large amounts of data for geographic regions are meteorology, disaster relief planning, prospecting and seismic survey.

Designing systems that support collaborating stakeholders in decision making is a complex task requiring development of functional display environments as well as appropriate visualizations for the situation at hand. In this paper, we first give an overview of related research in the field of collaborative visualization environments, which has led to technical solutions that support multiple 3D views but also general ways of partitioning visualizations for collaborative work. In our research we substantiate some research issues, discussed in the papers cited in the next section by presenting an experimental study aimed at investigating effects of visualizations of layer partitioning in collaborative spatial tasks.

2 Displays for Co-located Visualization

Many collaborative 3D environments support only one correctly rendered view for one user. Examples of such systems are the Responsive Workbench [1] and the CAVE [2]. With only one correctly rendered view it is problematic to discuss visualizations, since the same geometric feature is perceived differently from each user's perspective. In some collaborative 3D environments this problem has been solved by visualizing only views corresponding to those of a real physical environment. Examples of such environments are true volumetric displays like the Crossed-Beam Volumetric Displays [3] and Spatially Augmented Reality [4] where textures and illumination hints can be painted on physical models.

In our research we work with environments that in addition to realistic perspectives also can provide independently rendered views of the same co-located environment for multiple users. There are several different technical approaches capable of rendering independent views for multiple users [5,6,7,8,9,10,11,12]. The Two-user Responsive Workbench uses time multiplexing to generate two stereoscopic views [5]. The Virtual Showcase [6] is a system that uses half silvered mirrors together with a workbench that in addition of multiple independent views also can spatially augment real objects. IllusionHole uses a constrained spatial segmentation of a workbench display to provide independent views [7]. SCAPE uses head mounted projective displays to combine the inside-out views of a CAVE system with the outside-in views of workbench systems using retro-reflective material to avoid crosstalk between independent views [8]. Virtual Reality (VR) and Augmented Reality (AR) define a large field of interaction techniques that can provide independent views to individual collaborators [9,10,11]. In the context of co-located visualizations only optical see-through AR [9] is of interest, since it also allows for natural face-to-face collaboration between users.

The Multiple-Viewer Display Environment (Figure 1) produced by VAB (www.vargogat.se) and described in [12], provides four independent stereoscopic views by using pairwise projector polarization and a retro-projection screen that allows projected images to pass through the screen primarily along the optical axis. One objective for these technologies, capable of rendering independent views for multiple users, is to support view-dependent co-located visualizations.

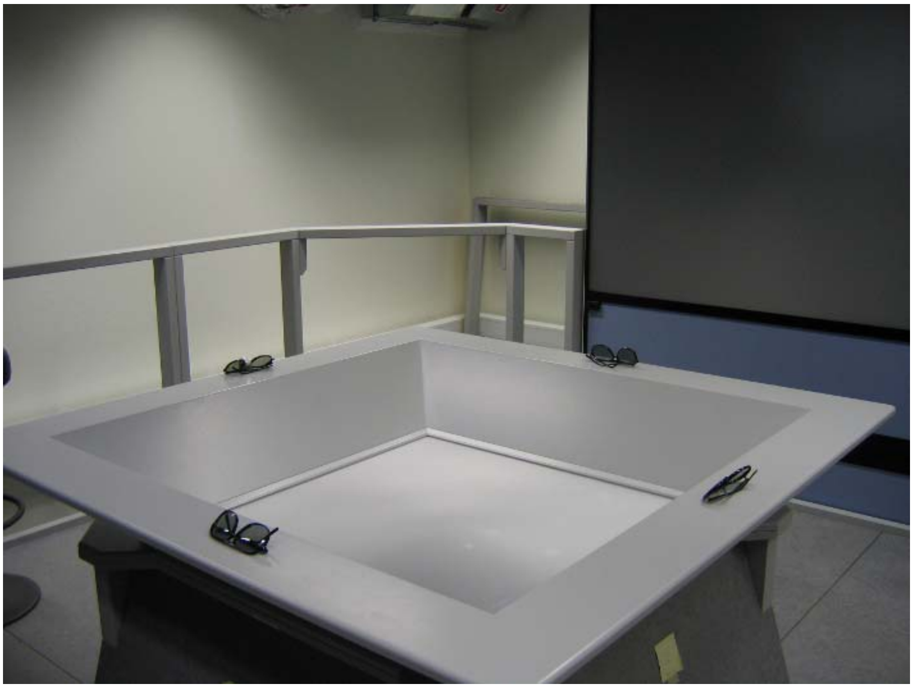


Fig. 1. The display used in the experiment can provide four independent stereoscopic views

3 View-Dependent Co-located Visualization

In the technical papers above many different terms have been used to describe how independent views for multiple users can support visualization in collaborative 3D environments. Agrawala et al. used the term *specialized views of a shared environment* to describe this form of visualization [5]. Bimber et al. used the term *view-dependent image presentation* [6]. Hua et al. called it *multiple independent views*. We have previously used the term *view-dependent co-located visualization* to emphasize that these kinds of visualizations share the same physical location [12,13], and we will consequently use it throughout the rest of this paper.

In view-dependent co-located visualizations, views with completely different content can be presented as well. However they do not support direct collaboration on the same data, due to the lack of common reference points. When at least one object is co-located in multiple views, it is possible to augment it with independent information. Agrawala et al. described this as "when displaying specialized information, the challenge is to ensure that the notion of a shared space is not lost" [5]. In their paper the authors make an assumption that the two-user responsive workbench can keep the users from being overwhelmed by extraneous information and help them to focus their attention on the most relevant details of the environment.

They demonstrate three scenarios of general strategies showing how specialized information can be presented.

The general strategies, according to [5], are:

- layer partitioning
- spatial partitioning
- private information

Layer partitioning is used to display only those layers of a visualization that each individual viewer is interested in as well as different abstraction levels of the same layer. Spatial partitioning is suggested for large display surfaces to decompose the viewing area into focus and non-focus regions, rendered in different resolutions. The last strategy, private information, is used to present information relevant only to one of the viewers. We have previously examined a fronto-parallel symbol visualization technique which does not fit in either of these strategies for specialized views [13]. The fronto-parallel visualization is neither layer or spatially partitioned nor is its main purpose to display private information. Its purpose is to enhance the readability of symbols and textual information of the collaborative 3D environment. Hua et al. have also suggested that "displaying multiple independent views offers the intriguing possibility of presenting different aspects or levels of detail of a shared environment in each user's view" [8]. It is therefore clear that the strategies suggested by Agrawala et al. need to be extended.

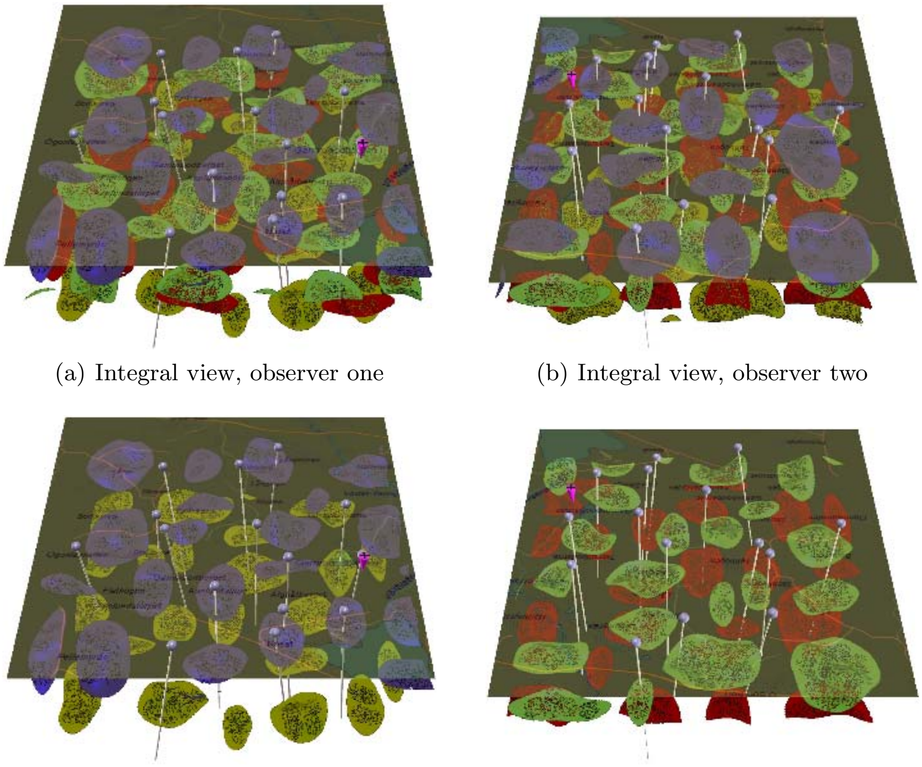
Although various ways of supporting view-dependent co-located visualization have been proposed in [5,6,7,8,12], only a few formal evaluations of their potential benefits have been carried out. In this paper we describe our initial research where we evaluate the efficiency of layer partitioning in a collaborative spatial task.

4 Experiment

4.1 Layer Partitioning

Integral and partitioned displays may well have an effect on the amount of communication needed between the collaborating users. At the same time, they also have an effect on the amount of visual clutter in each view. In an integral view, the visual clutter is larger for each user but the amount of communication needed is, at least potentially, smaller. In a partitioned view the effects are opposite. Do integral and partitioned views differ in efficiency, and if so, which alternative is more efficient?

To begin evaluating the efficiency of integral versus partitioned visualizations, we designed an experiment where pairs of observers (dyads) collaborated to solve a complex spatial task. See figure 2. A map was aligned with the surface of the display. Below the map the 3D visualization contained four layers with randomly distributed and irregularly shaped patches. From a reference point in the map the task was to identify the shortest distance to a borehole that intersects at least two specific layers.



(a) Integral view, observer one (b) Integral view, observer two

(c) Partitioned view, observer one (d) Partitioned view, observer two

Fig. 2. The screenshots of one stimulus in integral and partitioned condition illustrate varying degrees of visual clutter

In regard to a certain borehole in question, several spatial assessments are of interest:

– assessment of a qualitative spatial relation, if a borehole intersects or does not intersect
– assessment of relative spatial distance between entry point of borehole and the reference point in the 2D map
– identification of certain combinations of borehole intersections with the four layers

4.2 Stimuli

In the task a two-dimensional map was aligned with the surface of the display to provide a common frame of reference. See figure 2. The map was rendered transparently using alpha blending and in the same position as if a real physical map would have been placed on the surface of the display. Underneath that map, layers with spatially separated and irregularly shaped patches represented the spread of four different properties below ground. The patches were coded with distinctive colours. We used red and green as well as yellow and blue which are colour opponents located most distantly on the two chromatic axes according to the colour opponent theory [14]. Also, lightness levels were chosen such as to provide good luminance contrast.

The layers consisted of patches rendered in solid colors; however, in order to enhance spatial representation, they were perforated with a random stipple pattern, such that the layers underneath were partly visible through the upper layers. At fifteen randomly chosen positions in the map, boreholes were directed downward from the map surface. The angles of the boreholes could randomly vary up to ten degrees from the vertical axis. On the top of the boreholes a grey sphere indicated the origin of the borehole in the map. In each stimulus presentation the layers had different order and rotation. All boreholes had the same length, which was sufficiently long to pass through all layer depths. Each borehole could intersect none, one, or several layers. In the map layer a reference point was placed, modeled as a magenta cone with a cross. The subjects' task was to identify the closest borehole to the reference point satisfying a specific condition. The condition was that the borehole should intersect with patches at least from the green and yellow layers. Of the fifteen boreholes in each task at least three did match the intersection condition. To sustain an even level of difficulty, none of the three closest boreholes of the reference point was the correct answer. The assessment of relative distance between the reference point and a possibly correct borehole is done on the surface of the screen on the 2D map.
In this experiment we used two types of conditions:

- An *integral visualization*, where both users saw the entire visualization as shown in figure 2(a) and 2(b) at all times projected towards their respective point of view.
- A *partitioned visualization* as shown in figure 2(c) and 2(d), where each user only saw a subset of the visualization comprising the map along with the boreholes and only two out of the four layers. Each user either saw the green or the yellow layer.

4.3 Apparatus and Viewing Conditions

The display environment used in this experiment (Figure 1) was the same one as in our previously presented work [13,15]. It used four of eight possible simultaneous co-located views. The four views were used to provide independent stereoscopic bird's eye viewing metaphors to two observers. The display environment is rear projected with polarized filters on a square horizontal square screen area of 0.8 by 0.8 meters. The visible pixel resolution of each projection image on screen was 768 by 768 pixels. The software environment used to render the visualization is based on OpenGL and described in [12]. The display system was driven by a small cluster of four commercial off-the-shelf computers interconnected with a Giga-Ethernet local area network. The computers were equipped with Nvidia QuadroFX graphics cards. Head tracking was accomplished with an Ascension Flock of Birds system.

Two observers were standing on the opposite sides of the display environment and facing each other. Each observer simultaneously saw an independently rendered stereoscopic view provided by one pair of the four pairs of projectors. In total four of the eight projectors were used. The observers wore polarizing glasses with a Flock of Birds sensor attached, providing the head position. Movement was only restricted such that each observer could not move past their side of the edge of the display environment. Motion parallax not only enhanced the stereoscopic spatial illusion it was essential to solve the task by avoiding visual occlusion. Dim lighting was used in the room.

4.4 Experimental Design

The design of the study was a three-factor mixed design. The within factors were viewing condition, integral versus partitioned, and blocks of trials. The between factor was presentation order, if the dyad started with integral or partitioned visualization. The presentation order was counterbalanced.

4.5 Procedure

When the observers arrived at the experiment they were first asked about some background information, such as age, color vision, and if they had normal or corrected to normal vision.

During the experiment only one of the two participants in each condition could enter the answers using a keyboard and this participant was randomly chosen before the experiment started. They were given written and oral instructions of the experiment. The participants were told that accuracy was most important but that time also was crucial.

The trials were self-paced, and the observers controlled the screen by pressing the space bar when they wished a trial to begin. When a trial began the map and the layers were shown. The participants had to discuss and point to the map to come to an agreement on which borehole that was closest to the reference point and satisfied the intersection condition (yellow and green layers). When they had reached an answer, they pressed space bar again and the layers disappeared and a letter appeared next to every grey sphere located on the top of the boreholes. The answer was given by pressing the letter on the keyboard that coincided with the letter on the borehole. After this the screen turned black and the trial was over, the next trial began by pressing the space bar. Task time was measured between the first and second push on the space bar. This was done to capture only the time the participants were solving the task, not the time they spent on finding the right button to press on the keyboard.

The dyads performed two blocks of 15 trials, a total of 30 trials in each condition, integral and partitioned, resulting in a total sum of 60 trials for the dyads. Prior to each session of the different conditions a learning session of five pre-trials took place.

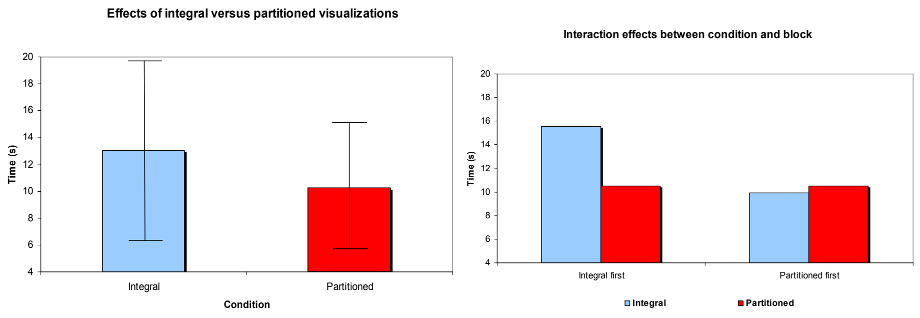
4.6 Observers

20 observers were divided into ten dyads. The subjects were all male and between 20 and 49 years old. The observers were either students at the Royal Institute of Technology (KTH) in Stockholm, Sweden, or students or staff of the Swedish National Defense College (SNDC), Stockholm. During a pre-test interview all participants stated that they had normal color vision and normal, or corrected to normal, vision.

They received a small compensation for taking part in the experiment and they had no prior knowledge of the purpose of the experiment of the specific hypotheses employed. The subjects were assigned to the dyads; in some of the resulting dyads the two subjects had a prior knowledge of each other.

5 Results

Errors were relatively scarce and approximately equal in number in the two main conditions. In the integral condition the dyads made, on average, 1.5 errors over the 30 trials and in the partitioned condition they made on average 1.3 errors per 30 trials.



(a) Mean times (b) Interaction effect

Fig. 3. To the left: Mean times and standard deviation to solve the task in integral and partitioned visualizations. To the right: Interaction effect between condition and presentation order.

As for the time data, the mean task time in the integral condition over the 30 trials was 13.1 seconds with a standard deviation of 6.6 seconds and in the partitioned condition 10.1 seconds with a standard deviation of 4.5 seconds. See figure 3(a). This means that the mean task time was 27 percent higher in the integral condition.

The time data were analyzed by means of an ANOVA. Since reaction time data typically are non-normally distributed, we employed a logarithmic transformation of these data before statistical testing. All calculations were conducted with a decision criterion of 0.05. The results from the ANOVA shows that there is a significant difference in favor of the partitioned visualization compared to the integral (F(1,8)= 14.30, $p<0.0054$). There is a significant effect of block, (F(1,8)= 10.54, $p<0.012$). There is also a interaction effect between condition and presentation order, depending on what condition the dyads started with, (F(1,8)= 13.49, $p<0.0063$). See figure 3(b). No other significant results came out of the ANOVA.

6 Discussion and Conclusion

The aim of this experimental study was to investigate effects of integral visualization as compared to partitioned visualization in collaborative spatial tasks. From the results of the ANOVA we find that the partitioned visualization led to significantly lower task performance times. That is, in spite of an additional overhead for communication between the subjects in the partitioned condition they performed faster compared to when viewing the integral visualization. One plausible explanation might be that, in the integral visualization, individuals had to change their head positions more frequently in order to resolve the intersections of the boreholes with the various layer patches. Obviously, in the presence of four layers at the same time, the two observers in a dyad had to resolve occlusions more often even though the visualization was perceived from two opposing directions. Another reason might be that in a more complex visualization subjects have more aspects to discuss and agree upon, whereas in a partitioned visualization each subject needs to accept the assessments of his respective peer.

In the learning session of the five pre-trials the subjects had the chance to develop a collaboration strategy. Finding this strategy in the trials after the pre-trials can be considered as part of the learning effect that was observed between successive blocks in both conditions. Since we were anticipating learning effects, we designed block sequences for the visualization conditions in order to study this effect.

A possible strategy, that probably most dyads used, was to solve the task in two stages. The first stage was to identify those borehole candidates which satisfied the specific intersection condition. In the second stage, the dyads determined the candidate with the shortest distance to the reference point. In the experiment we could observe that all the dyads used direct interaction, by pointing at the map in the display surface and discussing the visualization. It helped the subjects to remember the possible candidates while assessing distances within the map. This emphasizes the importance of view-dependent co-located visualizations as a means to provide spatially correct collaborative interaction for tasks similar to the one investigated in this experiment.

There was no difference in errors between the two conditions and error rate was low and similar for both. The subjects had been instructed that accuracy was most important but time was also crucial. This indicates that the subjects had adhered to the instructions given to them. We have no deeper insight if errors occurred due to incorrect intersection assessment or if they were due to incorrect estimation of the relative distances in the map plane.

The significant main effect of block (F(1,8)= 10.54, $p<0.012$) indicates that the participants successively became better during the experiment. It means that tasks in the experiment were not too difficult to learn. Another effect observed is the interaction between visualization condition and order of presentation (F(1,8)= 13.49, $p<0.0063$). Figure 3(b) shows that for the dyads starting with the partitioned visualization, task performance was the same as the integral visualization due to counteraction between learning effects and longer task times in the integral visualization. In contrast, for the groups starting with the integral visualization the performance was enhanced both by the learning effects and shorter task times for the partitioned visualization.

The results we obtained in this study hold for layer partitioning of visualizations into two subsets viewed and analyzed by two collaborating individuals. In this experiment the communication between the two peers was direct and they had a clear role assignment since only one subject controlled the keyboard. We assume that partitioning of visualizations for more than two users might lead to increased communication overhead which at some point might cancel out the gained effects of reducing visual clutter. Layer partitioning applies naturally to tasks which can be subdivided between observers such that each part is represented in a layer. “The approach is less useful for a user that needs information which is spread across many layers” [5].

In designing the task as we did, we did not aim at adapting it to some specific application area. Instead, we chose a task that aggregated two generic types of tasks, which are categorical assessment of qualitative features located in 3D (color and intersection) and of spatial features in 2D (distance). In combining these generic tasks to a more complex decision situation as tested in our study, we approached visualization scenarios more typical for real-world applications. Hence, we believe that the results in this paper are of relevance to a wider audience who intends to employ advanced displays for co-located visualizations in other fields of research such as geology, mining, infrastructure planning, and games.

Acknowledgements

The work presented in this paper is result of a sub-project funded by Swedish National Defense College (SNDC) in the framework of project AQUA. We specially thank Ulrik Spak at SNDC for discussing and helping with the experiment.

References

1. Krüger, W., Bohn, C.A., Fröhlich, B., Schüth, H., Strauss, W., Wesche, G.: The responsive workbench: A virtual work environment. Computer **28**(7) (1995) 42–48
2. Cruz-Neira, C., Sandin, D.J., DeFanti, T.A.: Surround-screen projection-based virtual reality: the design and implementation of the cave. In: SIGGRAPH ’93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM Press (1993) 135–142

3. Ebert, D., Bedwell, E., Maher, S., Smoliar, L., Downing, E.: Realizing 3d visualization using crossed-beam volumetric displays. Commun. ACM **42**(8) (1999) 100–107
4. Raskar, R., Welch, G., Chen, W.C.: Table-top spatially-augmented reality: Bringing physical models to life with projected imagery. In: IWAR '99: Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality, Washington, DC, USA, IEEE Computer Society (1999) 64
5. Agrawala, M., Beers, A.C., McDowall, I., Fröhlich, B., Bolas, M., Hanrahan, P.: The two-user responsive workbench: support for collaboration through individual views of a shared space. In: Proceedings of the 24th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co. (1997) 327–332
6. Bimber, O., Fröhlich, B., Schmalstieg, D., Encarnação, L.M.: The virtual showcase. IEEE Comput. Graph. Appl. **21**(6) (2001) 48–55
7. Kitamura, Y., Konishi, T., Yamamoto, S., Kishino, F.: Interactive stereoscopic display for three or more users. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, ACM Press (2001) 231–240
8. Hua, H., Brown, L.D., Gao, C.: Scape: supporting stereoscopic collaboration in augmented and projective environments. Computer Graphics and Applications, IEEE **24**(1) (2004) 66–75
9. Azuma, R.: A survey of augmented reality. Presence, Teleoperators and Virtual Environments **6**(4) (1997) 355–385
10. Snowdon, D., Greenhalgh, C., Benford, S.: What you see is not what i see: Subjectivity in virtual environments. In: Framework for Immersive Virtual Enviroments (FIVE'95), QMW University of London (1995)
11. Smith, G., Mariani, J.: Using subjective views to enhance 3d applications. In: VRST '97: Proceedings of the ACM symposium on Virtual reality software and technology, New York, NY, USA, ACM Press (1997) 139–146
12. Pettersson, L.W., Spak, U., Seipel, S.: Collaborative 3d visualizations of geo-spatial information for command and control. In: Proceedings of SIGRAD 2004. (2004) 41–47
13. Pettersson, L.W., Lind, M., Spak, U., Seipel, S.: Visualizations of symbols in a horizontal multiple viewer 3d display environment. In: Information Visualisation, 2005. Proceedings. Ninth International Conference on. (2005) 357 – 362
14. Ware, C.: Information Visualization: Perception for Design. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2004)
15. Forsell, C., Seipel, S., Lind, M.: Simple 3d glyphs for spatial multivariate data. In: INFOVIS '05: Proceedings of the 2005 IEEE Symposium on Information Visualization, Washington, DC, USA, IEEE Computer Society (2005) 119–124

GPU-Based Active Contour Segmentation Using Gradient Vector Flow

Zhiyu He and Falko Kuester

Calit2 Center of GRAVITY
University of California, Irvine
zhe@uci.edu, fkuester@uci.edu

Abstract. One fundamental step for image-related research is to obtain an accurate segmentation. Among the available techniques, the active contour algorithm has emerged as an efficient approach towards image segmentation. By progressively adjusting a reference curve using combination of external and internal force computed from the image, feature edges can be identified. The Gradient Vector Flow (GVF) is one efficient external force calculation for the active contour and a GPU-centric implementation of the algorithm is presented in this paper. Since the internal SIMD architecture of the GPU enables parallel computing, General Purpose GPU (GPGPU) based processing can be applied to improve the speed of the GVF active contour for large images. Results of our experiments show the potential of GPGPU in the area of image segmentation and the potential of the GPU as a powerful co-processor to traditional CPU computational tasks.

1 Introduction

In the area of image based analysis and its related applications, segmentation is, in many cases, the starting point for further processing. The segmentation algorithm may provide the foundation for further processing, such as identifying features or objects that subsequently are used for the reconstruction of 3D models. Among many existing segmentation algorithms, the active contour technique or *snake* [1] is an algorithm that uses an *external force* and an *internal force* to progressively fit a closed curve to edges, boundaries or other features of interest specified via gradient. The snake has been widely used in areas such as biomedical image analysis and further enhanced for specific problem domains. For example, Xu and Prince [2] proposed a better way of calculating the external force of the curve. This improved snake algorithm is called *Gradient Vector Flow* (GVF) snake and has two advantages over the original snake algorithm: (1) it is less sensitive to initialization and (2) it can move into boundary concavities. This paper introduces a hardware accelerated technique for gradient vector flow computation, utilizing the vertex and fragment units on today's graphics processing units. Most mid-range GPUs now have a SIMD architecture and deep parallel processing capabilities on the vertex and fragment units [3], which can be used as a very efficient co-processor that can take over some of the computation tasks otherwise handled by the CPU. These computation tasks are not limited to graphics and visualization but may also include general purpose computation. This paper is organized as follows: Section 2 introduces background and prior work done in related areas. Section 3 gives an overview of GPGPU based processing and how it is related to this research. Section 4 describes the GPU implementation of the GVF snake algorithm. Section 5 provides a performance and test results.

2 Related Work

The segmentation by active contour or snake algorithm can be found in combination with many image related applications. Kass et al. [1], introduced the snake algorithm, which uses an external force and an internal force to conform the contour to certain features in the image. The external force is calculated from the image and the internal force is derived from the contour itself. The corresponding curve is defined by:

$$X_t(s,t) = \alpha X''(s,t) - \beta X''''(s,t) - \nabla E_{ext} \tag{1}$$

where $X_t(s,t)$ is the curve that represents the snake at time t and $X(s) = [x(s), y(s)], s \in [0,1]$ is the parametric curve, X'' and X'''' are the second and fourth order derivatives, α and β are constants that defines the internal forces. ∇E_{ext} is the external force. The GVF snake introduced by Xu and Prince [2] improves the above by introducing a new external force. The revised dynamic snake function can then be formulated as:

$$X_t(s,t) = \alpha X''(s,t) - \beta X''''(s,t) + V \tag{2}$$

where V stands for the new static external force field called *gradient vector flow* (GVF). Zimmer et al. [4] applied the algorithm to video tracking for the quantitative analysis of cell dynamics. Ding et al. [5] described a volumetric CT data segmentation that is based on application of GVF snake to 2D CT slices. Vidholm et al. [6] introduced a virtual reality system for the visualization of volume data combined with force-feedback. GVF snake segmentation of the data was used for visual augmentation and control of the haptic device. Some of the GPU processing and bandwidth characteristics can outpace that of CPUs, which make it appealing to convert processing extensive algorithms to the GPGPU domain if their nature is compatible. For example, Rumpf et al. [7] introduced a level-set based segmentation that was leveraging GPU capabilities. Despite of the advantages of the level-set segmentation, the implementation was still limited by the graphics hardware available at that time and therefore is not completely GPU centric. Kondratieva et al. [8] described a real-time computing and visualization technique for diffusion tensor images, which achieves both visual and speed improvements over traditional CPU realization. Fan et al. [9] built a computing cluster based on GPU to achieve greater parallel processing power. Kipfer et al. [10] implemented a fluid dynamics simulation engine on the GPU, which leverages the GPU to avoid I/O bottlenecks and improves performance. Fatahalian et al. [11] implemented an efficient matrix multiplication algorithm on the GPU. GPU based computation is not limited to the above mentioned areas and can be expand to many other areas compatible with the SIMD architecture.

3 GPU and GPGPU

Recent GPUs demonstrate enormous potential for scientific computing tasks in the form of General Purpose GPU-based processing (GPGPU). In particular, memory bandwidth and instructions per second highlight potential benefits. For instance, the Nvidia Geforce 6800 graphics chip can process 600 Million vertices/sec and has a fill rate of 6.4 billion pixels/sec, while the Geforce 7800 series can almost double that performance. Galoppo et al. [14] reported that the 6800 could achieve 2.5 billion instructions per second for division, which compare to 6.7 billion for a Pentium4 3.2GHZ CPU. Kilgariff and Fernando [3] demonstrated that the GPU Memory Interface of the Geforce 6800 series can reach 35 GB/sec, which compares well against the 6.4 GB/sec of the CPU Memory Interface for a 800 MHz Front-Side Bus. Besides these, the GPU has a very different architecture and processing stream than the CPU. The GPU processing model can be decomposed into several stages ([15]). Data goes from the CPU to GPU through system bus. On the GPU, it goes from vertex buffer, the vertex processor, rasterization and finally gets to the fragment processor. One important feature of GPU is its SIMD architecture that naturally supports parallel processing. Most computation tasks on GPU are parallelized as illustrated in Fig.1. For example, the Geforce 6800 supports 6 vertex units and 16 fragment units. And each unit can process 4 components (RGBA or xyzw) in parallel.

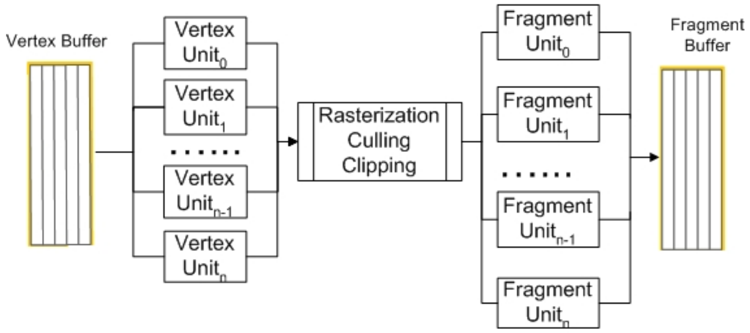


Fig. 1. The parallel nature of GPU

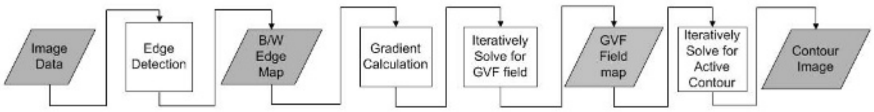


Fig. 2. The process of GVF Snake

However, it is important to carefully consider strengths and weaknesses of GPU-based techniques. First of all, although the GPU has excellent computational power, the majority of graphics cards are still limited to 16bit floating point precisions. This means, in many cases, the traditional implementations of algorithms will be subjected to a loss in precision when migrated directly onto the GPU. One solution is to use 2 components of texture unit to store one 32bit float number. With more graphics cards supporting the 32bit floating point textures, this problem will be reduced in the near future. However, it is still very important to find a balance between the precision and speed because the 32bit floating point data lead to nearly half the speed of the 16bit precision data as reported in [16].

Secondly, while the bandwidth on the CPU or on the GPU alone can be enormous, the bus I/O between CPU and GPU can sometimes become a bottleneck. On the Geforce 6800 card, the PCI Express$\times$16 inteface provides 8 GB/sec throughput while the on-board bandwidth for the GPU is 35 GB/sec. Therefore, it is worthwhile to optimize the code for fewer I/O on the GPU.

Third, the data structure should fit to the platform architecture. When implementing an algorithm on the GPU, it is important to consider its SIMD architecture. The data should be independent from each other, and random access of data such as a linked-list should be avoided if all possible.

Shader Model 3.0 and the OpenGL 2.0 standard provide a means to resolve the problems mentioned this far. For example, multiple rendering target could save rendering passes by using a single input texture to generate multiple output textures. In addition, Frame Buffer Objects (FBOs) greatly improve the speed by saving I/O between GPU and CPU. The vertex texturing functionality allows the texture to be used as a data array. In support of hardware-based processing, different high-level languages were created, such as CG [17] and [18], which supports most features for Shader Model 3.0. HLSL [19] and the OpenGL Shading Language [20] are also such languages.

4 Gradient Vector Flow Snake Implementation on GPU

Equation (2) describes the GVF-based snake function, which introduced the V term for the gradient vector flow. V can be defined as a vector field $V(x,y) = [u(x,y), v(x,y)]$ that minimizes the energy function:

$$\varepsilon = \int\int \mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + |\nabla f|^2 \, |V - \nabla f|^2 \, dxdy \tag{3}$$

where $f(x,y)$ is an edge map of the original image, ∇f is its gradient map and μ is a constant that represents the level of noise. To solve Equation (3) for the $V(x,y)$, u and v need to be treated as functions of time by solving the following equations:

$$u_t(x,y,t) = \mu\nabla^2 u(x,y,t) - b(x,y)u(x,y,t) + c^1(x,y)$$
$$v_t(x,y,t) = \mu\nabla^2 v(x,y,t) - b(x,y)v(x,y,t) + c^2(x,y)$$

where:

$$b(x,y) = f_x(x,y)^2 + f_y(x,y)^2,\ c^1(x,y) = b(x,y)f_x(x,y),\ c^2(x,y) = b(x,y)f_y(x,y)$$

and ∇^2 is the laplacian operator. This can be numerically expressed as:

$$u_t = \frac{1}{\Delta t}(u_{i,j}^{n+1} - u_{i,j}^{n}), v_t = \frac{1}{\Delta t}(v_{i,j}^{n+1} - v_{i,j}^{n})$$

$$\nabla^2 u = \frac{1}{\Delta x \Delta y}(u_{i+1,j} + u_{i,j+1} + u_{i-1,j} + u_{i,j-1} - 4u_{i,j})$$

$$\nabla^2 v = \frac{1}{\Delta x \Delta y}(v_{i+1,j} + v_{i,j+1} + v_{i-1,j} + v_{i,j-1} - 4v_{i,j})$$

By substituting the above variables into the equations for $u_t(x,y,t)$ and $v_t(x,y,t)$, an iterative solution to the GVF field can be obtained.

The GVF snake algorithm is composed of two parts: (1) the pre-computing of the GVF field and (2) the iterative solution of the snake function. Both parts has the temporal and spatial locality. At any single time step, the $u(x,y)$ and $v(x,y)$ only involves 4 of its neighboring points and 1 previous time step. The general flow for this algorithm is illustrated in Figure 2. First, the input image is converted to greyscale and an edge detection filter is applied to obtain the edge map. Subsequently, a second shader is used to obtain the gradient map and generates three contants for every pixel, namely $b(x,y)$,$c^1(x,y)$ and $c^2(x,y)$. The multiple rendering target technique is then used to generate and store the results in two seperate textures, one for the gradient and the other for the constants. Most current GPUs only support 16bit floating point precision with values clamped to the range of $[0.0, 1.0]$. Therefore, we store the data using a *packing* scheme. The gradient dx and dy are stored in R and G components and the B and A components save a flag number identifying how the dx and dy are stored. In this particular case, the dx is stored as is if $|dx| \geq 0.01$, otherwise, as $-1/\ln dx$. Similar *packing* is performed on the three constants.

After these preparation steps, the iterative GVF field calculation can start. The iterative computation on GPU can be mapped to a so-called *ping-pong* scheme using the FBO(*frame buffer objects*). Each FBO can be bound to four framebuffers, namely, COLOR0 through COLOR3. This is illustrated in following pseudo code:

```
Src_Buffer = COLOR0;  Dst_Buffer = COLOR1;
while(counter<Number)
{Attach Dst_Buffer as DrawBuffer; Src_Buffer = input for fragment shader;
 Draw the texture;
 Swap the Src_Buffer and Dst_Buffer;
 Increase counter;}
```

The resultant GVF field is stored in one framebuffer and is used as input parameter to the snake process fragment shader. The fragment shader for the snake process involves solving a linear system:

$$\overline{A} * \overline{X_t} = \gamma * \overline{X_{t-1}} + \kappa * \overline{V} \tag{4}$$

where $\overline{V}$ is the GVF field and $\overline{X_t}$ is the snake contour at t time, γ and κ are constants and $\overline{A}$ is a constant matrix. Note that solving the above linear system not only requires an inverse of the matrix, but also brings in the violation of the spatial locality. While the first problem can be addressed on the GPU [14], the second problem will dramatically decrease the efficiency of the GPU implementation because it results in extensive amount of I/O for texel fetch operations. However, $\overline{A}$ is very similar to a symmetric band matrix and is positive definite, with the exception that the upper right corner and lower left corner of the matrix is not zero. It can be expressed as Equation (5a).

$$\begin{array}{cccccccccc}
c & b & a & 0 & 0 & 0 & \ldots\ldots 0 & a & b \\
b & c & b & a & 0 & 0 & 0 & \ldots\ldots 0 & a \\
a & b & c & b & a & 0 & 0 & \ldots\ldots 0 & 0 \\
0 & a & b & c & b & a & 0 & \ldots\ldots 0 & 0 \\
\multicolumn{10}{c}{\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots} \\
\multicolumn{10}{c}{\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots} \\
a & 0 & 0 & 0 & \ldots\ldots 0 & a & b & c & b \\
b & a & 0 & 0 & 0 & \ldots\ldots 0 & a & b & c
\end{array} \quad [5a.] \quad
\begin{array}{cccccccccc}
C & B & A & 0 & 0 & 0 & \ldots\ldots 0 & A & B \\
B & C & B & A & 0 & 0 & 0 & \ldots\ldots 0 & A \\
A & B & C & B & A & 0 & 0 & \ldots\ldots 0 & 0 \\
0 & A & B & C & B & A & 0 & \ldots\ldots 0 & 0 \\
\multicolumn{10}{c}{\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots} \\
\multicolumn{10}{c}{\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots} \\
A & 0 & 0 & 0 & \ldots\ldots 0 & A & B & C & B \\
B & A & 0 & 0 & 0 & \ldots\ldots 0 & A & B & C
\end{array} \quad [5b.] \qquad (5)$$

where a, b are positive constant values that are much smaller than 1 and c is around 1. The major diagonal has the value of c, the second major diagonal has the value of b and the next one is a. Furthermore, its inverse matrix is of similar type. Because most of the numbers in $\tilde{A}$ are far smaller than 10^{-6} after the third diagonal, an approximation of $\tilde{A}$ can be given as Equation (5b). where:

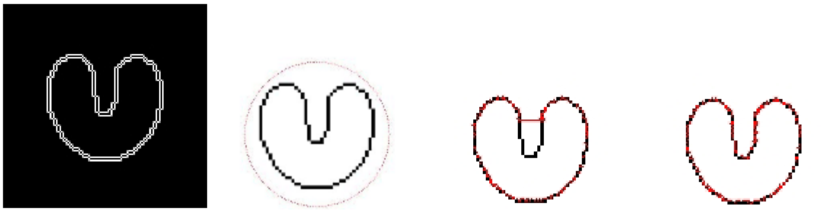
$$\begin{array}{c}
A = c(b^2 - ac)/\,[X]\,,\ B = \frac{-b[X]-bc(b^2-ac)(a-c)}{(c^2-b^2)[X]} \\
C = \frac{c[X]-c(b^2-ac)(a-b^2)}{(c^2-b^2)[X]},\ [X] = (c^2-a^2)(c^2-b^2) - b^2(c-a)^2
\end{array} \qquad (6)$$

By inserting the above equations to Equation (4), a discrete solution for the snake contour can be obtained and implemented as a shader program: $x_i^t = \sum_{k=i-2}^{i+2} M_{i,k}^{t-1} v_k$, $i \in [1, n]$, where n is the number of points in the snake contour and $M_{i,k}^{t-1}$ is the row i, column k element of the matrix multiplication result of $\tilde{A}$ and the column matrix of x_i^{t-1} and $v_k = \gamma x_k^{t-1} + \kappa f_k$.

5 Experiment Results

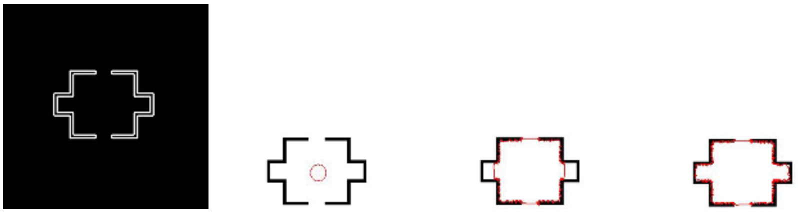
All tests were performed on a laptop PC with a 2.4GHZ Pentium processor, 1GB RAM and a Geforce Go6800 card with 128MB on-board graphics memory. Fig.3(a) shows the edge map for the U-shape image, followed by the initial curve in Fig.3(b), a partial result after 150 iterations in Fig.3(c) and the final result in Fig.3(d). The result shows that the GVF snake algorithm can contract to concave shapes where ordinary snake could not. Fig.4 shows a non-continuous room model results. These two data sets were modeled after the ones by Xu and Prince [2] to provide a better comparison. Fig.5 is an MRI brain scan which shows the algorithm can be applied to real-world data with non-uniform background and concave shape. A set of scans of human shoulder was studied at levels of image ranging from 128×128 to 1024×1024 pixel resolution in order to evaluate the scalability of the algorithm and pinpoint performance tradeoffs. The results for computations on the above data are provided in Fig.6.

In Fig.6, the GVF field calculation was performed both on the GPU and CPU, and it includes the edge detection stage for the GPU. For the snake algorithm,



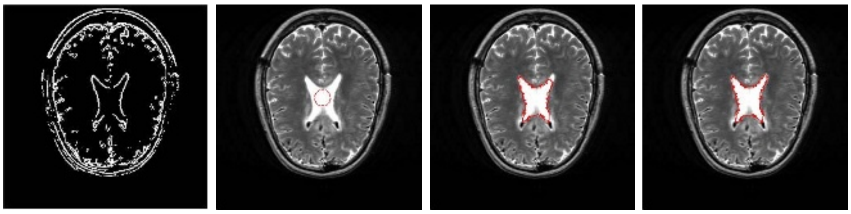
(a) Edge map (b) Initial circle (c) Partial result (d) Final result

Fig. 3. U-shape 256 × 256 pixel



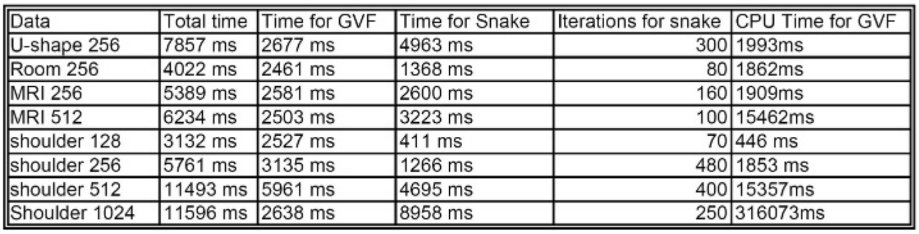
(a) Edge map (b) Initial circle (c) Partial result (d) Final result

Fig. 4. Room 256 × 256 pixel



(a) Edge map (b) Initial circle (c) Partial result (d) Final result

Fig. 5. MRI 256 × 256 pixel



Data	Total time	Time for GVF	Time for Snake	Iterations for snake	CPU Time for GVF
U-shape 256	7857 ms	2677 ms	4963 ms	300	1993ms
Room 256	4022 ms	2461 ms	1368 ms	80	1862ms
MRI 256	5389 ms	2581 ms	2600 ms	160	1909ms
MRI 512	6234 ms	2503 ms	3223 ms	100	15462ms
shoulder 128	3132 ms	2527 ms	411 ms	70	446 ms
shoulder 256	5761 ms	3135 ms	1266 ms	480	1853 ms
shoulder 512	11493 ms	5961 ms	4695 ms	400	15357ms
Shoulder 1024	11596 ms	2638 ms	8958 ms	250	316073ms

Fig. 6. Benchmark on test images

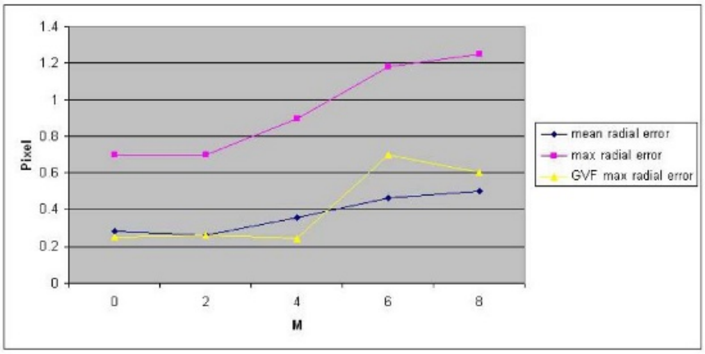
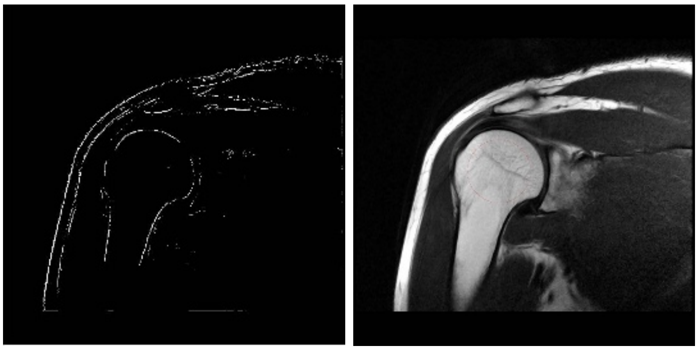
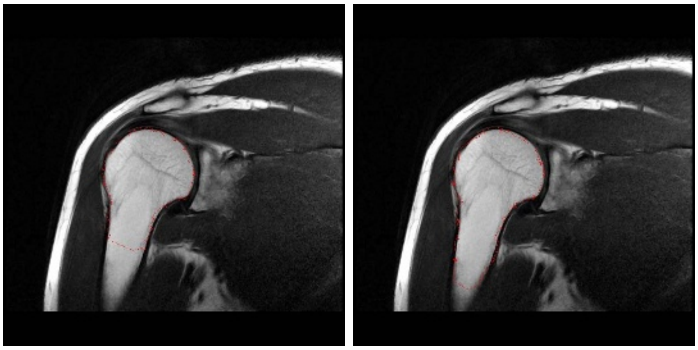


Fig. 7. MRE comparison



(a) Edge map (b) Initial circle



(c) Partial result (d) Final result

Fig. 8. Shoulder 1024 × 1024 pixel

it can be observed that the speed performance complexity can be expressed as $O(n^2k)$, where n^2 is the size of the texture and k is the number of iterations. This means that newer graphics cards with more texture memory will be able

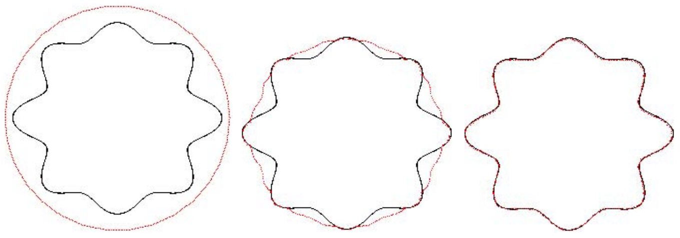

(a) Initial circle (b) Partial result (c) Final result

Fig. 9. Spine

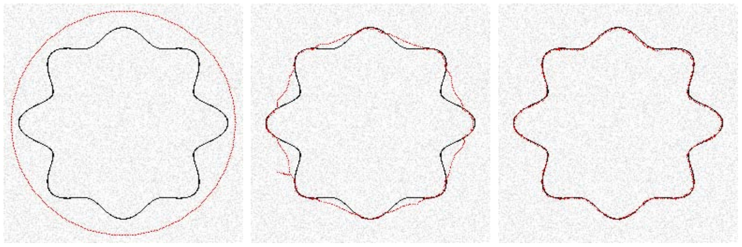

(a) Initial circle (b) Partial result (c) Final result

Fig. 10. Spine with Gaussian noise

to efficiently process larger images. For images of 256 × 256, the CPU is about 20% faster than the GPU. But for 512 × 512 image, the GPU technique starts to outperform the CPU by 4 times. Therefore, the parallel capability of GPU computing shows its advantages on larger images. Each individual GPU fragment or vertex processor is lower than the CPU. However, with the increase in data size, the GPU parallel pipeline becomes more efficient and greatly outpaced the CPU. Another observation is that texture I/O may become the bottleneck. For example, the texture fetch for the snake shader is more than two times that of the GVF shader and so the snake shader is 50% slower.

One test case is studied to analyze the accuracy of the GPU technique (Fig.9 and Fig.10). This test case uses simple harmonic curves given by: $r = a + b \cos m\theta + c$, where a, b, c are constant values and by varying the m, a set of curves can be obtained. Each image is 256×256 and we used $m = 0, 2, 4, 6, 8$. The measure of error is MRE(mean radial error), which is the mean distance in the radial direction between the final active contour and the harmonic curve. Fig.7 shows the MRE result. The blue line shows the MRE, the red line shows the maximum radial error as the worst case scenario and the yellow line shows the maximum radial error from a CPU implementation of improved GVF algorithm as stated in [21]. As we can see, the performance of CPU implementation generally has better accuracy. The reason for the performance gap is the difference in the precision of floating point data. Nonetheless, the GPU implementation still achieves a good overall accuracy and the mean errors are within sub-pixel level.

Fig.10 shows the robustness of the GPU technique with the addition of gaussian noise. The image with noise has an MRE of 0.5 while the clean image is 0.35.

6 Conclusion

A hardware accelerated gradient vector flow algorithm for image segmentation was presented. The algorithm utilizes the fragment and texture units of the GPU. A set of test cases was presented and evaluated comparing CPU and GPU results. In addition, some new features of GPGPU are exploited and some important issues involved in porting algorithms onto the GPU are specified, which provides a foundation for further exploration in this algorithm.

References

1. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. International Journal of Computer Vision. **1** (1988) 321–331
2. Xu, C., Prince, J.L.: Snakes, shapes, and gradient vector flow. IEEE Transaction on Image Proccessing. **7** (1998) 359–369
3. Kilgariff, E., Fernando, R.: The geforce 6 series gpu architecture. In: GPU Gems 2 : Programming Techniques for High-Performance Graphics and General-Purpose Computation. (2005) 471–493
4. Zimmer, C., Labruyere, E., Meas-Yedid, V., Guillen, N., Olivo-Marin, J.: Segmentation and tracking of migrating cells in videomicroscopy with parametric active contours: A tool for cell-based drug testing. IEEE Transaction. on Medical Imaging **21** (2002) 1212–1221
5. Ding, F., Leow, W., Wang, S.: Segmentation of 3d ct volume images using a single 2d atlas. In: Lecture Notes in Computer Science. Volume 3765., Springer (2005) 459–468
6. Vidholm, E., Nystrom, I.: Haptic volume rendering based on gradient vector flow. In: Proceedings of Swedish symposium on image analysis (SSBA'05). (2005) 97–100
7. Rumpf, M., Strzodka, R.: Level set segmentation in graphics hardware. In: Proceedings of the 2001 International Conference on Image Processing. Volume 3. (2001) 1103–1106
8. Kondratieva, P., Krüger, J., Westermann, R.: The application of gpu particle tracing to diffusion tensor field visualization. In: Proceedings IEEE Visualization 2005(Vis'05). (2005)
9. Fan, Z., Qiu, F., Kaufman, A., Yoakum-Stover, S.: Gpu cluster for high performance computing. In: Proceedings of the 2004 ACM/IEEE conference on Supercomputing (SC'04). (2004) 47
10. Kipfer, P., Segal, M., Westermann, R.: Uberflow: a gpu-based particle engine. In: Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware (HWWS'04). (2004) 115–122
11. Fatahalian, K., Sugerman, J., Hanrahan, P.: Understanding the efficiency of gpu algorithms for matrix-matrix multiplication. In: Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware (HWWS'04). (2004) 133–137
12. Lefohn, A.E., Kniss, J., Hansen, C., Whitaker, R.: Interactive deformation and visualization of level set surfaces using graphics hardware. In: Proceedings of the 14th IEEE Visualization(VIS'03). (2003) 75–82

13. Yang, R., Welch, G., Bishop, G.: Real-time consensus-based scene reconstruction using commodity graphics hardware. In: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications (PG'02). (2002) 225
14. Galoppo, N., Govindaraju, N., Henson, M., Manocha, D.: Lu-gpu: Efficient algorithms for solving dense linear systems on graphics hardware. In: Proceedings of the 2005 ACM/IEEE conference on Supercomputing (SC'05). (2005) 3–3
15. Lefohn, A.: Gpu memory model overview. In: Proceedings of the ACM Siggraph 2004. (2004)
16. Govindaraju, N., Raghuvanshi, N., Henson, M., Manocha, D.: A cache-efficient sorting algorithm for database and data mining computations using graphics processors. In: UNC Tech. Report. (2005)
17. NVidia: The cg toolkit. In: http://developer.nvidia.com/object/cg_toolkit.html. NVidia Corp. (2005)
18. gpgpu.org: General-purpose computation on gpus. (2005)
19. Microsoft: Hlsl shaders. In: http://msdn.microsoft.com. Microsoft Inc. (2004)
20. OpenGL: Opengl shading language. In: http://www.opengl.org/documentation/oglsl.html. OpenGL.org (2005)
21. Xu, C., Prince, J.L.: Generalized gradient vector flow external forces for active contours. Signal Processing — An International Journal **71** (1998) 131–139

Active Single Landmark Based Global Localization of Autonomous Mobile Robots

Abdul Bais[1,3], Robert Sablatnig[2], Jason Gu[3], and Stefan Mahlknecht[1]

[1] Institute of Computer Technology
Vienna University of Technology
Vienna, Austria
bais@ict.tuwien.ac.at*

[2] Pattern Recognition and Image Processing Group
Institute of Computer Aided Automation
Vienna University of Technology
Vienna, Austria
sab@prip.tuwien.ac.at

[3] Robotics Research Laboratory
Department of Electrical and Computer Engineering
Dalhousie University
Halifax, Canada
jason.gu@dal.ca

Abstract. This paper presents landmark based global self-localization of autonomous mobile robots in a known but highly dynamic environment. The algorithm is based on range estimation to naturally occurring distinct features as it is not possible to modify the environment with special navigational aids. These features are sparse in our application domain and are frequently occluded by other robots. To enable the robot to estimate its absolute position with respect to a single landmark it is equipped with dead-reckoning sensors in addition to the stereo vision system mounted on a rotating head. The pivoted stereo vision system of the robot enables it to measure range and use bi/trilateration based methods as they require fewer landmarks compared to angle based triangulation. Further reduction of landmarks is achieved when robot orientation is estimated independently. Simulation results are presented which illustrate the performance of our algorithm.

1 Introduction

In an application where multiple robots are working on a common global task, knowledge of position of individual robots turns out to be a very basic requirement for its execution. A successful global strategy can be devised if the robot knows its own position and those of other robots with whom it has to interact. A simple solution for robot position estimation would be to start from a known location and track the robot position locally using methods such as odometry or

* Supported by Higher Education Commission of Pakistan.

inertial navigation [1,2]. These methods have proven to be very efficient and provide good short term position estimates but suffer from unbounded error growth due to integration of minute measurements to obtain the final estimate [3]. The failure of local methods to track the robot position over extended periods of time and the requirement of an initial estimate makes global position estimation a necessity. In global position estimation external sensors are used to sense the environment and calculate position [4]. In this case a robot has no information other than that it is somewhere on a map.

If it is not allowed to modify the robot environment with artificial landmarks or active beacons, the robot has to extract naturally occurring distinctive features in its environment and to estimate its position with respect to them. Such features, scarce in our application domain, are frequently occluded by other robots for longer durations. This makes simultaneous acquisition of multiple landmarks troublesome and hence the localization algorithm should be based on as few landmarks as possible. If the robot is only able to measure angles between landmarks then a minimum of three distinct landmarks are required to triangulate the robot position on a planar surface [5, 6]. Whereas when range measurements are available this requirement drops to two if ordering of the landmarks with respect to the robot is possible [7].

There have been approaches to maximize the chance of simultaneous acquisition of multiple landmarks using omni-directional cameras with viewing angle of $360\,^{\circ}$ [8,9]. These approaches suffer from high cost of the mirror, low resolution of the camera, and requirement of an additional space to fit the mirror and the camera. With single frontal cameras one can have high resolution but the field of view is limited [10, 11]. Additionally, range measurement using single image is too erroneous and the approach cannot be used at all times [12].

To provide the missing range information to camera images, some researchers fuse information from range and intensity sensors. There have been approaches combining laser range finder with single frontal cameras [12, 13] and omni-directional cameras [14]. In order to get advantages of range measurements, we propose a stereo vision system. To overcome the problems with the limited field of view of the directional cameras the stereo cameras are mounted on a pivoted head. This is an aid in exploring the robot environment for features and can be made small enough to fit within given dimensions.

However, even with the pivoted stereo head we experience difficulties with simultaneous acquisition of two distinct landmarks. Therefore, we extend our existing algorithm so that the absolute position of the robot may be calculated with respect to one landmark where the robot orientation is estimated with another independent source i.e. with a compass. To the best of our knowledge no one has applied this method in a similar setup.

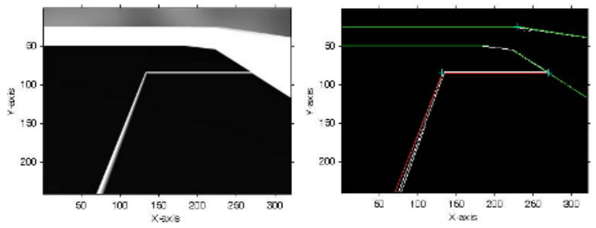
In this paper we focus on performance analysis of single landmark global self-localization. Landmark extraction and range estimation is done with the stereo vision system. The robot environment is simple but highly dynamic consisting of visual landmarks i.e. lines, corners, junctions, line intersections and color transitions [15]. The test bed for our algorithm is a soccer playing robot called

Tinyphoon(www.tinyphoon.com) [16]. The balance of the paper is organized as follows: Section 2 discusses position estimation using single landmark. Uncertainty analysis of the method is carried out in Section 3. Experimental results are presented in Section 4 and the paper is finally concluded in Section 5.

2 Landmark-Based Localization

Landmarks are distinct features that a robot can recognize from its sensory input. In general, they have a fixed and known position, relative to which a robot can estimate its position [17]. Input data may be of range or bearing type. This leads to two different techniques, trilateration and triangulation, respectively. Trilateration is the determination of a robot's position based on distance measurements to landmarks, whereas, in triangulation, bearing to different landmarks in the environment is used [17].

The environment consist of color transitions, corners, junctions and line intersections as landmarks. Corners, junctions and line intersection are detected using semantic interpretation of line segments extracted using gradient based Hough transform [15] (see Fig. 1). Whereas, color segmentation of the camera images is used to detect specific color transitions in the environment [7]. In this study we use only color transitions as landmarks.



(a) Left camera image (b) Detected features superimposed over the edge map

Fig. 1. Line based landmarks for self-localization

2.1 Position Estimation Using Single Landmark

We assume that the robot's motion is two dimensional where pose of the robot has 3 degrees of freedom i.e. $\mathbf{p} = [x\ y\ \theta]^T$. The global coordinate system is represented by x and y axis, whereas the robot coordinate system by X and Y. As can be seen in Fig. 2, the robot position is constrained to a circle C' when it detects a distinct landmark point $p_l = [x_l\ y_l]^T$ in the global coordinate system and measures its range $r = \sqrt{X^2 + Y^2}$. Assuming identical cameras, parallel image planes and aligned epipolar lines the landmark point ($[X\ Y\ Z]^T$) in the camera coordinate system can be related to its projection in the left and right image using (1) [18].

$$\begin{aligned} X &= x_c + \frac{fb}{u_l - u_r} \\ Y &= \frac{-b}{2} \frac{u_l + u_r - 2o_u}{u_l - u_r} \\ Z &= \frac{-b(v_l - o_v)}{u_l - u_r} \end{aligned} \tag{1}$$

where $[u_l \; v_l]^T$ and $[u_r \; v_r]^T$ are the coordinates of the landmark point in the left and right image, $[o_u \; o_v]^T$ is the image center point, b is the baseline of the stereo vision system, f focal length of both cameras and x_c is the distance from the center of the robot to the cameras.

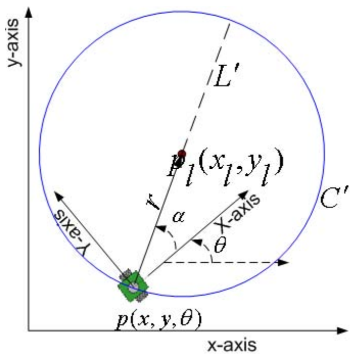


Fig. 2. Single landmark based localization: robot position constrained to the intersection of line and circle

Now, as shown in Fig. 2, if in addition to measuring $[X \; Y]^T$ the orientation of the robot is known, its position is further constrained to the intersection of circle C' and line L'. The two intersection points are given by (2).

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_l \pm \frac{\sqrt{X^2+Y^2}}{\sqrt{1+m^2}} \\ y_l \pm \frac{m\sqrt{X^2+Y^2}}{1+m^2} \end{bmatrix} \tag{2}$$

where slope of the line $m = \tan(\theta + \alpha)$ and $\alpha = atan2(\frac{Y}{X})$.

In our case the line segment always originates at the robot location and terminates at the landmark. This information may be used to resolve the ambiguity between the two candidate positions. For the case shown in Fig. 2 the robot's orientation and location of the landmark in robot's coordinate system is such that it can only be on the lower left side of the landmark. The use of this information reduces (2) to (3).

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_l - \sqrt{X^2+Y^2}\cos(atan2(\frac{Y}{X}) + \theta) \\ y_l - \sqrt{X^2+Y^2}\cos(atan2(\frac{Y}{X}) + \theta) \end{bmatrix} \tag{3}$$

From this discussion it is clear that range measurement with respect to a distinct landmark and knowledge of the robot's orientation is sufficient to calculate its absolute location. The stereo vision system is used to extract description of landmarks and calculate their range. This description is used to identify landmarks in the global map of the environment. To measure the robot orientation it is equipped with a yaw rate sensor. It is most accurate ($< 3\,^\circ$) when the robot is not moving and there are no other robots or magnetic objects in its close proximity [16].

3 Uncertainty Analysis

Error free measurements will result in perfect localization. However, measurements are never perfect and errors in distance and angle estimates can vary significantly. In addition to measurement errors there could be error in landmark identification and matching with the world map [19]. Due to error in measured image coordinates and orientation estimate, the robot will be somewhere in the shaded area shown in Fig. 3. The extent of this uncertainty area is dependent on the amount of error.

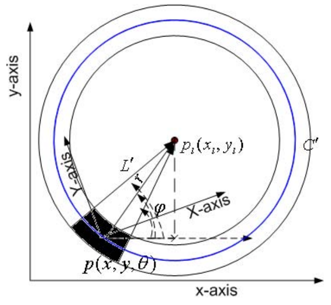


Fig. 3. With error in range and angle the robot position is constrained to the shaded area determined by the intersection of the thickened circle and the line

As errors make it impossible to estimate the exact position, it is required to associate some sort of uncertainty measure with the robot position. We assume perfect landmark location and its matching with the world map. We further assume that u_l and u_r are corrupted by an additive Gaussian noise with zero mean and variance σ^2_{uu}, similarly, error in the orientation estimate is Gaussian with zero mean and variance $\sigma^2_{\theta\theta}$.

Error in u_l and u_r is propagated to X and Y by (1), whereas error in X, Y and θ is propagated to the estimated position by (3). As the transformations (1) and (3) are non-linear the resulting error distribution is not Gaussian. We use first order approximation of these equations for error propagation. According to our experiments this captures the uncertainty adequately. The uncertainty in $[u_l\ u_r]^T$ propagated into $[X\ Y]^T$ by (1) is given by the following expression [20].

$$\mathbf{\Sigma_{IR}} = \mathbf{J_{IR}\Sigma_I J_{IR}^T} \tag{4}$$

where $\mathbf{\Sigma_I} = \sigma_{uu}^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ is the covariance of $[u_l \ u_r]^T$ and $\mathbf{J_{IR}}$ is the Jacobian of (1) with respect to $[u_l \ u_r]^T$. Simplification of (4) results in a 2×2 matrix which is updated to a 3×3 matrix to accommodate uncertainty in θ. The updated covariance matrix is given by the following expression.

$$\mathbf{\Sigma_{IR}} = \begin{bmatrix} 2\frac{b^2}{d^4} f^2 \sigma_{uu}^2 & -\frac{b^2}{d^4} f \sigma_{uu}^2 (u_r + u_l - 2o_u) & 0 \\ -\frac{b^2}{d^4} f \sigma_{uu}^2 (u_r + u_l - 2o_u) & \frac{b^2}{d^4} \sigma_{uu}^2 ((u_r - o_u)^2 + (u_l - o_u)^2) & 0 \\ 0 & 0 & \sigma_{\theta\theta}^2 \end{bmatrix} \tag{5}$$

where $d = u_l - u_r$ is the stereo disparity. θ is estimated independently, hence error in θ is independent of error in X and Y.

Finally, error in $[X \ Y \ \theta]^T$ is propagated to the robot position $\mathbf{p} = [x \ y]^T$ by (3). Again using first order approximation we derive the expressions for the covariance matrix of $\mathbf{p}$

$$\mathbf{\Sigma_{RW}} = \mathbf{J_{RW}} \mathbf{\Sigma_{IR}} \mathbf{J_{RW}^T} \tag{6}$$

where

$$\mathbf{J_{RW}} = \begin{bmatrix} J_{xX} & J_{xY} & J_{x\theta} \\ J_{yX} & J_{yY} & J_{y\theta} \end{bmatrix} \tag{7}$$

simplification of (6) results in the following

$$\mathbf{\Sigma_{RW}} = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_{yy}^2 \end{bmatrix} \tag{8}$$

where

$$\begin{aligned} \sigma_{xx}^2 &= J_{xX}^2 \sigma_{XX}^2 + 2 J_{xX} J_{xY} \sigma_{XY}^2 + J_{xY}^2 \sigma_{YY}^2 + J_{x\theta}^2 \sigma_{\theta\theta}^2 \\ \sigma_{xy}^2 &= J_{xX} J_{yX} \sigma_{XX}^2 + J_{xY} J_{yX} \sigma_{XY}^2 + J_{xX} J_{yY} \\ &\quad \sigma_{XY}^2 + J_{yY} J_{xY} \sigma_{YY}^2 + J_{y\theta} J_{x\theta} \sigma_{\theta\theta}^2 \\ \sigma_{yy}^2 &= J_{yX}^2 \sigma_{XX}^2 + 2 J_{yX} J_{yY} \sigma_{XY}^2 + J_{yY}^2 \sigma_{YY}^2 + J_{y\theta}^2 \sigma_{\theta\theta}^2 \end{aligned} \tag{9}$$

in (9) $J_{xX} = -\frac{cX}{r} - \frac{sY}{r}$, $J_{xY} = \frac{sX}{r} - \frac{CY}{R}$, $J_{x\theta} = sr$, $J_{yX} = \frac{cY}{r} - \frac{sX}{r}$, $J_{yY} = -\frac{sY}{r} - \frac{cX}{r}$, $J_{y\theta} = -CR$ are elements of the Jacobian matrix of (3) with respect to $[X \ Y \ \theta]^T$ and $s = \sin(atan2(\frac{Y}{X}) + \theta)$, $c = \cos(atan2(\frac{Y}{X}) + \theta)$, $r = \sqrt{X^2 + Y^2}$.

4 Experimental Results

We investigate the performance of our algorithm in simulation. The robot is currently under development and real world data shall be available in near future. In all of these experiments image resolution and the stereo baseline was set at 320×240 pixels and $30mm$ respectively. We have conducted 28 trials where each one has 100 steps. These trials are further grouped into six categories: motion

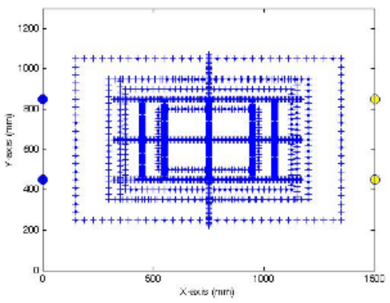
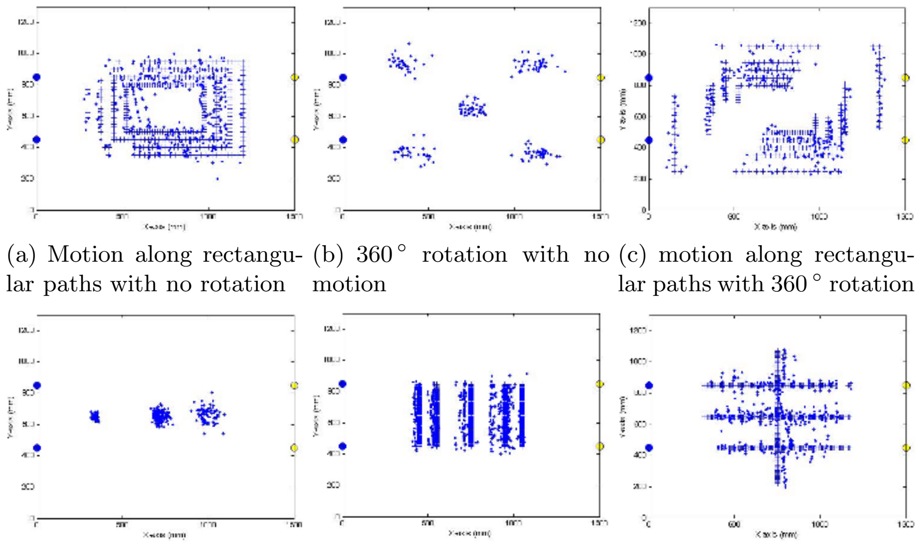


Fig. 4. Actual locations from where position was calculated



(a) Motion along rectangular paths with no rotation (b) 360 ° rotation with no motion (c) motion along rectangular paths with 360 ° rotation

(d) Only 90 °rotation with no motion (e) motion along linear paths with 20 ° rotation (f) motion along linear paths with no rotation

Fig. 5. Normally distributed error with mean 0 and variance 5 °added to the robot orientation

along rectangular paths with no rotation, 360 °rotation with no motion, motion along rectangular paths with 360 °rotation, 90 °or 180 °rotation with no motion, motion along linear paths with 20 °rotation and motion along linear paths with no rotation. At every step the robot is taking images of its environment, searches for color transitions and calculates its position if it finds one. In these experiments only a single shot localization method is used. The algorithm does not incorporate any kind of tracking of landmarks or of its position.

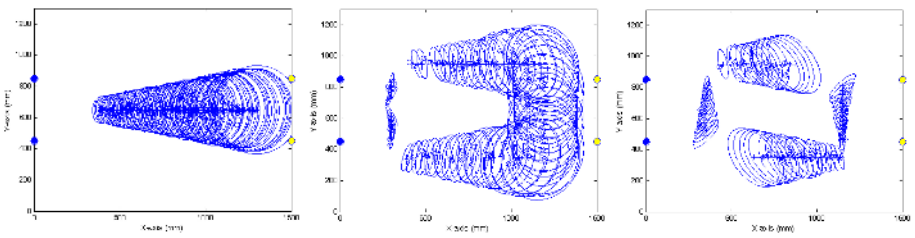
Fig. 4 shows the path followed by the robot. Locations where images were taken and searched for color transitions are shown as dots (·). However, depending on the instantaneous pose of the robot, it may not find any landmark

Table 1. Error in x and y with perfect orientation estimate

	μ_1	σ_1	μ_2	σ_2	μ_3	σ_3	μ_4	σ_4	μ_5	σ_5	μ_6	σ_6
δx	41.68	43.42	45.64	40.98	38.92	34.48	34.66	27.88	41.04	27.69	43.90	27,61
δy	9.48	11.17	20.48	16.78	20.33	17.01	12.90	10.15	7.12	24.69	4.31	5.4

Table 2. Error in x and y where error in robot orientation is normally distributed with zero mean and 5 ° variance

	μ_1	σ_1	μ_2	σ_2	μ_3	σ_3	μ_4	σ_4	μ_5	σ_5	μ_6	σ_6
δx	41.51	44.05	48.73	42.19	41.37	34.98	35.53	28.58	41.63	28	44.15	28.27
δy	22.30	25.05	29.07	24.44	27.97	23.35	23.94	21.49	24.04	30.25	24.13	19.78



(a) Line motion: The uncertainty decreases as the robot gets closer to the landmark

(b) Motion only: Following a rectangular path with the fixed orientation

(c) Motion and rotation: Following a rectangular path with 360 °of rotation in 100 steps

Fig. 6. Uncertainty analysis

therefore its pose is estimated only at limited locations shown as plus (+) superimposed on the dots(·).

The first category consisting of four trials is shown in Fig. 5(a). The robot follows a rectangular paths of different dimension around the field with its orientation fixed at 0 ° or 180 °. Fig. 5(b) illustrates a 360 °rotation-only category consisting of 5 trials. Here the robot is placed at five locations: near the four corners and at the center of the field. Motion along rectangular paths of different dimension and rotation of 360 °in each trial is shown in Fig. 5(c). This category consists of 5 trials. Fig. 5(d) shows the fourth category that consist of 4 trials. In this case the robot's position is fixed at one point and it completes a rotation of 90 °or 180 °in a trial. Similarly, motion along linear paths and 20 °rotation in small steps is shown in Fig. 5(e). Fig. 5(f) illustrate the last category consisting of 5 trials. Here the robot is following linear paths but its orientation is fixed along the x-axis.

Statistical results for error in position are shown in Table 1 and Table 2 for all the six categories as discussed above. The subscripts 1 to 6 represents the category. For each category mean (μ) and standard deviation (σ) for the absolute error in x and y is shown in both the tables. Table 1 shows the results where perfect orientation of the robot is used, whereas, Table 2 shows the case where the robot orientation estimation is in error. Errors δx and δy in Table 1 and Table 2 are expressed in millimeters.

Fig. 6 shows 50% uncertainty ellipses for each estimate of the robot position based on the discussion presented in Section 3. For this experiment the σ^2_{uu} and $\sigma^2_{\theta\theta}$ values were chosen to be 0.5 and $3\,^\circ$ respectively. Error in robot orientation was set to a maximum of $\pm 8\,^\circ$ with a variance of $5\,^\circ$. There is a substantial increase in the uncertainty as the robot gets further from the landmark point. Analysis of error in landmark location and the effect of dropping the higher order terms in the Taylor series expansion is the subject of our current research.

5 Conclusion

In this paper we have presented our investigation of self-localization using range measurement to a single landmark. Simulation results show that this method can successfully localize robots in an environment with scarce landmarks. For this study only color transitions are used as landmarks. Extraction of color patches is very efficient as only $N/16$ pixels are tested to determine the rectangular boundaries around them (if any), N being the total number of pixels [7]. We have used just a single shot localization and have not incorporated any kind of temporal redundancy. The robot pose could be refined once an estimate is available. Currently we are working on methods for efficient interpretation of landmarks other than color transitions, tracking of landmarks, tracking robot position with local sensors and information fusion.

References

1. Chung, H., Ojeda, L., Borenstein, J.: Accurate mobile robot dead-reckoning with a precision-calibrated fiber-optic gyroscope. IEEE Transactions on Robotics and Automation **17** (2001) 80 – 84
2. Komoriya, K., Oyama, E.: Position estimation of a mobile robot using optical fiber gyroscope (ofg). In: International Conference on Intelligent Robots and Systems (IROS'94), Munich, Germany (1994) 143–149
3. Borenstein, J.: Experimental results from internal odometry error correction with the omnimate mobile robot. IEEE Transactions on Robotics and Automation **14** (1998) 963 – 969
4. Arsenio, A., Ribeiro, M.: Absolute localization of mobile robots using natural landmarks. In: Proceedings IEEE International Conference on Electronics, Circuits and Systems. Volume 2. (1998) 483 – 486
5. Sugihara, K.: Some location problems for robot navigation using a single camera. Computer Vision, Graphics, and Image Processing **42** (1988) 112–129
6. Yuen, D.C.K., MacDonald, B.A.: Vision-based localization algorithm based on landmark matching, triangulation, reconstruction, and comparison. IEEE Transactions on Robotics **21** (2005) 217 – 226

7. Bais, A., Sablatnig, R.: Landmark based global self-localization of mobile soccer robots. In Narayanan, P.J., Nayar, S.K., Shum, H.Y., eds.: Computer Vision ACCV 2006: 7th Asian Conference on Computer Vision. Volume 3852 of Lecture Notes in Computer Science., Hyderabad, India, Springer-Verlag GmbH (2006) 842 – 851
8. Steinbauer, G., Bischof, H.: Illumination insensitive robot self-localization using panoramic eigenspaces. In: RoboCup 2004. Volume 3276 of LNAI. (2005) 84 – 96
9. Ji, J., Indiveri, G., Ploeger, P., Bredenfeld, A.: An omni-vision based self-localization method for soccer robot. In: IEEE symposium on Intelligent Vehicles (IV'03), Columbus, Ohio, USA (2003)
10. Bandlow, T., Klupsch, M., Hanek, R., Schmitt, T.: Fast image segmentation, object recognition and localization in a robocup scenario. In: RoboCup-99: Robot Soccer World Cup III. (1999) 174–185
11. Choi, W., Ryu, C., Kim, H.: Navigation of a mobile robot using mono-vision and mono-audition. In: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (IEEE SMC '99). Volume 4. (1999) 686–691
12. Nickerson, S.B., Jasiobedzki, P., Wilkes, D., Jenkin, M., Milios, E., Tsotsos, J., Jepson, A., Bains, O.N.: The ark project: Autonomous mobile robots for known industrial environments. Robotics and Autonomous Systems **25** (1998) 83–104
13. Weber, J., Franken, L., Jorg, K.W., Puttkamer, E.: Reference scan matching for global self-localization. Robotics and Autonomous Systems **40** (2002) 99–110
14. Clerentin, A., Delahoche, L., Pegard, C., Brassart, E.: A localization method based on two omnidirectional perception systems cooperation. In: Proceedings of the IEEE International Conference on Robotics and Automation. (2000) 1219–1224
15. Bais, A., Sablatnig, R., Novak, G.: Line-based landmark recognition for self-localization of soccer robots. In: IEEE International Conference on Emerging Technologies (ICET '05), Islamabad, Pakistan (2005) 132–137
16. Novak, G., Mahlknecht, S.: TINYPHOON a tiny autonomous mobile robot. In: IEEE International Symposium on Industrial Electronics (ISIE' 05). (2005) 1533–1538
17. Borenstein, J., Everett, H.R., Feng, L.: Navigating Mobile Robots: Systems and Techniques. A. K. Peters, Ltd. (1996)
18. Trucco, E., Verri, A.: Introductory TEchniques for 3-D Computer Vision. Prentice Hall, Upper Saddle River, NJ, USA (1998)
19. Sutherland, K.T., B.Thompson, W.: Inexact navigation. In: IEEE International Conference on Robotics and Automation (ICRA' 93). (1993) 1–7
20. Clarke, J.C.: Modelling uncertainty: A primer. Technical Report 2161/98, University of Oxford, Dept. Engineering Science (1998)

Iterative Estimation of 3D Transformations for Object Alignment

Tao Wang and Anup Basu

Department of Computing Science, Univ. of Alberta, Edmonton, AB T6G 2E8, Canada

Abstract. An Iterative Estimation Algorithm (IEA) of 3D transformations between two objects is presented in this paper. Skeletons of the 3D objects are extracted using a fully parallel thinning technique, feature point pairs (land markers) are extracted from skeletons automatically with a heuristic rule, and a least squares method and an iterative approach are applied to estimate the 3D transformation matrix. The algorithm has three advantages. First of all, no initial transformation matrix is needed. Secondly, user interaction is not required for identifying the land markers. Thirdly, the time complexity of this algorithm is polynomial. Experiments show that this method works quite well with high accuracy when the translations and rotation angles are small, even when noise exists in the data.

1 Introduction

3D alignment or registration algorithms have many applications in medical image processing. Consider two objects *O1* and *O2*, such that *O2=M*O1*, where *M* is the 3D transformation matrix. In this context, the objective of 3D alignment or registration is to estimate the 3D transformation matrix *M*.

Surveys [1-2] of 3D alignment methods are available in the literature. Since the previous decades, a lot of 3D alignment algorithms, including *land marker based algorithm* [3], have been proposed. This paper focuses on land marker based algorithms. The mean square distance (MSD) [3] is used as the metric. Point correspondence [4], which relates a land marker with its counterpart, is a crucial step for algorithms in this category. The advantage of this technique is that the set of land markers is relatively sparse compared to the original 3D object, so that the optimization process is relatively fast. However, this technique has some disadvantages. First of all, some algorithms, for instance, the ICP [3] algorithm, needs to know the initial transformation matrix. Secondly, user interaction is usually required for identifying the land markers. Last but not the least, the complexity of point correspondence [4] is non-polynomial since it is a combinatorial optimization problem.

In this paper, an automatic Iterative Estimation Algorithm (IEA) is proposed. It has three advantages: (i) no initial transformation matrix is needed; (ii) user interaction is not required for identifying the land markers; (iii) the complexity of this algorithm is polynomial. The disadvantage of this algorithm is that it does not guarantee achieving global optimization.

We applied IEA in 3D medical image processing. We are interested in defining and tracking volume changes of airways caused by surgery, which increases volume. Doctors need to track the changes of airways for administering effective treatments. In our application, doctors take MRI scans of patients and save them in the DICOM [5] format. A commercial software ScanIP/FE [6] is used to segment the airways and remove noise semi-automatically. The skeletons are created to depict the segmented airway using a fully parallel 3D thinning algorithm developed by us. Then, we applied a heuristic algorithm to automatically extract feature point pairs (land markers) from skeletons. Since false point correspondence is inevitable, we use IEA to remove some point pairs and use the remainder point pairs to achieve smaller MSD. The whole procedure has 5 main steps: segmentation, noise removal, thinning, finding feature point pairs and iterative transformation estimation.

In the following sections, the algorithm is described in detail. In Section 2, we introduce the data collection and segmentation procedures. Section 3 will briefly discuss how to artificially create data for comparison. The fully parallel 3D thinning algorithm and the skeletonization process is summarized in Section 4. Section 5 focuses on the acquisition of feature point pairs (land markers), and the IEA. Results of experiments are presented in Section 6, before the work is concluded in Section 7.

2 Data Collection, Segmentation and Noise Removal

A small portion of the upper airway is scanned for a child as an MRI image and saved in the DICOM format, which is the industry standard for MRI. ScanIP/FE is used to segment airways, remove noise, and semi-automatically create an airway model of 3D images. A *3D image* is a mapping that assigns the value of 0 or 1 to each point in the 3D space. Points having the value of 1 are called *black (object)* points, while 0's are called *white (background)* ones. Black points form objects of the image. Our test data has about 37,000 object points. Fig. 1 shows some image slices and the segmented airway (in red). The 3D models from different viewpoints are displayed in Fig. 2.

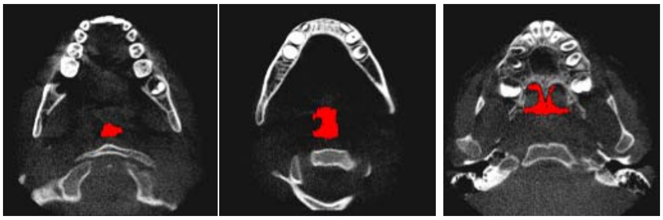


Fig. 1. Some image slices and segmented airway (in red). (*Left*) The first image slice. (*Middle*) A center slice. (*Right*) The last slice.

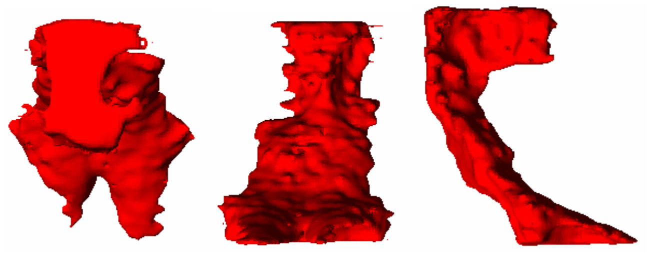


Fig. 2. 3D airway model from 3 different points of view

3 Data Creation for Estimation

To verify our method, we artificially created some airway models for estimation. We applied 3D homogeneous transformations [7] to the 3D image. We use (*alpha, beta, gama, dx, dy, dz*) to represent the rotation angles and translations for x-, y- and z-axis of the Homogeneous Matrices. We also added some random noise to the data to create models for comparison.

4 Fully Parallel 3D Thinning Algorithm and Skeletonization

3D thinning for medial lines or surface approximation is a useful approach for many potential applications. The approach has been extensively researched in the last decade [8]. In the experiments, we applied our fully parallel 3D thinning algorithm to extract the skeletons from 3D images after a noise removal step. Our method is an improvement on Ma and Sonka's algorithm [8]. The algorithm in [8] has a number of advantages; however, we found that it cannot preserve connectivity of 3D objects. Chaturvedi *et al.* [9] also found this problem. We improved [8] by changing some masks in Class D to preserve connectivity. The skeletons are used to represent the airway models. The skeletons have about 500 object points, a large reduction from the 37,000 object points.

Fig. 3 (*Left*) and (*Middle*) show the skeletons of the original model and the artificially created model with (*alpha, beta, gama, dx, dy, dz*) = (1, 1, 1, 0, 0, 0). We notice that:

1. The thinning algorithm is sensitive to rotations. Even when the rotation angles are very small, the two skeletons look quite different.
2. The thinning algorithm cannot provide unit-width structures. We can see that some regions on the skeletons are quite dense.

However, we will show that these two drawbacks do little harm to the estimation of 3D transformations. Thus, we can still estimate the transformations precisely based on the thinning algorithm.

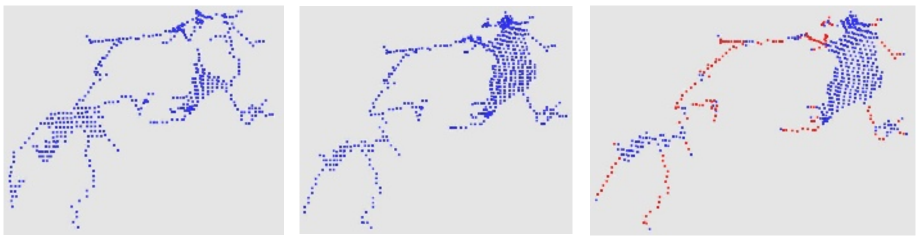


Fig. 3. (*Left*) The skeleton of the original model. (*Middle*) The skeleton of the transformed model with (*alpha, beta, gama, dx, dy, dz*) = (1, 1, 1, 0, 0, 0). (*Right*) Line points (in red).

5 Feature Point Pair Acquisition and the IEA

Land marker based estimation algorithms have three main disadvantages as we discuss before. In this section, an Iterative Estimation Algorithm (IEA) is proposed to solve these problems. The disadvantage of this algorithm is that it does not guarantee global optimization.

Definition 1. Connectivity number
*If P is an object point in a 3D image the connectivity number of P is the number of object points except itself in P's (3*3*3) neighborhood.*

In Fig. 4 (*Left*), the connectivity number of P is 4.

Definition 2. Line point
A line point is an object point with connectivity number equal to 2.

In Fig. 3 (*Right*), the line points are displayed in red.

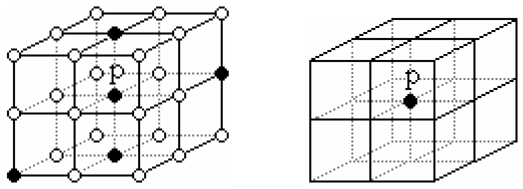


Fig. 4. (*Left*) Connectivity number of point P is 4. A " ● " is an object point, A " ○ " is a background point. (*Right*) Point P and its 8 sub-neighborhood.

Definition 3. Feature point (land marker) candidate
*A feature point (land marker) candidate is a line point with at least one non- line point in its (3 * 3 * 3) neighborhood.*

Definition 4. Feature point pair (land marker)
*I1 and I2 are two 3D images of the same object. I2=M*I1, where M is a 3D transformation matrix. A feature point pair (land marker) contains two object points P1 and P2, where P1 is on the skeleton of 3D image I1 and P2 is on the skeleton of 3D image I2, and P2=M*P1.*

5.1 The Overdetermined Linear System

In general, *M* is a 4 * 4 Homogeneous Transformation Matrix. For point $P1 = (x1\ y1\ z1\ 1)^T$ in 3D space, $P2 = M * P1 = (x2\ y2\ z2\ 1)^T$. The goal of our work is to estimate the 16 variables of matrix M in certain conditions. Since we need to calculate 16 variables, mathematically, we need 16 equations for this linear system. For each feature point pair, we can have 4 equations. So we need 4 feature point pairs in total to solve this problem. However, in most cases in our experiment, we can find more than 4 feature point pairs. This is a typical case of an over-determined linear system. An over-determined linear system can be solved following well-established methods [10].

5.2 The Heuristic Rule

In our experiments, a heuristic rule is used to identify the feature point pairs. It requires two points in a feature point pair to have same "configurations" in its 8 sub-neighborhood. That is, the local topologies of two points in a feature point pair should be same. We used an adaptive method to search for the best neighborhood scale. Fig. 4 (*Right*) shows a point p and its 8 sub-neighborhood.

Heuristic Rule
If two feature point candidates on different skeletons have the same number of object points and the same number of background points in all 8 sub-neighborhoods, these two points form a feature point pair.

5.3 The Iterative Estimation Method

Feature point correspondences is an open problem in Computer Vision. No general method can solve it in polynomial time. There are mainly two difficulties. Firstly, it is a combinatorial optimization problem. The time complexity is very high because of the large search space. Secondly, point correspondence is not one-one mapping. Some points, which are called “outliers” [4, 11], may not have counterparts because of occlusions, out of image transformations and errors in feature selection.

A brief survey of previous work is available in the literature [4]. For several decades, researchers have proposed a number of point correspondence algorithms [4, 11]. Point correspondence algorithms have many applications [4, 12-13] including image alignment. The point correspondence algorithms are classified into two categories, *globally optimal* and *non-globally optimal.* A globally optimal algorithm [4] guarantees global optimality. However, the time complexity is non-polynomial. A non-globally optimal algorithm [11] is faster than a globally optimal algorithm. However, it does not guarantee global optimality. Another restriction is that some unwanted assumptions and constrains [11] are often required to detect the outliers.

The IEA is a non-globally optimal algorithm. The complexity of this algorithm is polynomial. No user interaction, initial transformation knowledge and outlier detection are required. The motivation behind our work is that we realized that the characteristics of our application, as well as many other medical imaging applications, are: the number of feature point pairs is large, user interaction is unwanted, initial transformation knowledge is not easy to know and the outlier detection, which requires some unwanted assumptions, constrains, and is not very reliable, should be avoided. In these cases, we need a polynomial algorithm that requires no user interaction, no initial transformation information, implicitly removes outliers without any unwanted assumptions and constrains.

We notice that incorrect point correspondences and incorrect outlier detection negatively affect the estimation accuracy. However, we also notice that incorrect point correspondences and incorrect outlier detection are unavoidable. Therefore, in our algorithm, we use a very simple Heuristic Rule to do the point correspondences. There are some incorrect point correspondences and some outliers exist in the feature point pairs set. We do not know which point pair is correctly related and which point pair is incorrectly related. Since the incorrect point correspondences and incorrect outlier detection negatively affect the estimation, i.e., the MSD is large, we iteratively remove one point pair from the whole set and test if the new MSD is smaller. If the new MSD is smaller than the old MSD, we accept this removal. Otherwise, we undo the removal. The iteration stops when a preset MSD threshold is achieved or only 4 point-pairs remain. This approach is very simple and easy to implement. It removes the incorrect point correspondences and removes outliers implicitly. The pseudo code of the Iterative Estimation Algorithm (IEA) is as follows:

```
INPUT: 3D image $I_1$ and 3D image $I_2$, feature point candidate sets $C_1$ and $C_2$, and the MSD threshold $MSD_{THRESHOLD}$
OUTPUT: 3D transformation matrix $M$ and the $MSD$
1. Read $I_1$, $I_2$, $C_1$ and $C_2$
2. Select feature point pairs $P = (P_1, P_2)$ with the Heuristic Rule, where $P_1 \subseteq C_1$ and $P_2 \subseteq C_2$. Denote by $N_{FP}$ the number of feature point pairs.
3. Initialize the mean square distance $MSD$ and the transformation matrix $M$ with a predefined value. If $N_{FP} < 4$, then go to Step 10.
4. Use least square method to solve the over-determined linear system with $P$, and get the 3D transformation matrix $M$. Create a new 3D image $I_3$, where $I_3=M *I_1$
5. Calculate the $MSD$ between $I_2$ and $I_3$.
6. If $MSD < MSD_{THRESHOLD}$, then go to Step 10.
FOR ($INDEX = 0$; $INDEX < N_{FP}$; $INDEX$ ++)
    7. Remove a feature point pair with index = $INDEX$ from $P$ to create feature point pairs $P'$, calculate the 3D transformation matrix $M'$, create a new 3D image $I_3'$, where $I_3' = M' * I_1$ and calculate mean square distance $MSD'$ between $I_2$ and $I_3'$.
    8. If $MSD<MSD_{THRESHOLD}$ or $N_{FP}<4$, then $M=M'$ and go to Step 10.
    9. If $MSD' < MSD$, then $MSD = MSD'$, $P = P'$. Otherwise, undo the remove operation in Step 6.
END FOR
10. Output the $M$ and $MSD$.
```

The time complexity of the IEA is polynomial. Step 1, 3, 6 and 10 of the IEA has constant complexity. Step 4 and 5 has polynomial complexity. The complexity of Step 2 is $O(N_{C1} * N_{C2})$, where N_{C1} and N_{C2} are the number of feature point candidates in C_1 and C_2. The complexity of Step 7-9 is $O(N_{FP}) * (T_{LS}+T_{MSD})$, where T_{LS} is the complexity of least square method and T_{MSD} is the complexity of calculating MSD. Both of T_{LS} and T_{MSD} are polynomial. So the total time complexity of IEA is polynomial.

To test our approach, we integrate it into our previous framework of 3D transformation estimation. Experiments in Section 6 show that this approach with IEA can estimate 3D transformation with higher accuracy. The new framework is described as follows:

```
INPUT: 3D image $I_1$ and $I_2$, the MSD threshold $MSD_{THRESHOLD}$,
and the range of neighborhood searching $NB_{MIN}$ and $NB_{MAX}$.
OUTPUT: 3D transformation matrix between $I_1$ and $I_2$
Read $I_1$ and $I_2$, create their skeletons $S_1$ and $S_2$, Init $M_{EST}$
and $MSD_{EST}$ with a predefined value
FOR ($NB = NB_{MIN}$; $NB <= NB_{MAX}$; $NB$ += 2)
1. Calculate the feature point candidate sets $C_1$ from
$S_1$, $C_2$ from $S_2$, within current neighborhood scale $NB$
2. Call the IEA to get $M$ and $MSD$
3. If $MSD < MSD_{THRESHOLD}$, then $M_{EST} = M$ and go to Step 5.
4. If $MSD_{EST} > MSD$, then $MSD_{EST} = MSD$ and $M_{EST} = M$.
END FOR
5. Output the estimated 3D transformation matrix $M_{EST}$
```

In our experiments, $NB_{MIN} = 3$, $NB_{MAX} = 21$ and $MSD_{THRESHOLD} = 1.0$. The experiments and the results are described in Section 6.

6 Experimental Results

We first applied our new method with IEA to estimate the 3D translations along the x-, y- and z-axes. All the translations are positive integers in [0, 5]. Fig. 5 shows the MSD for 3D translations. Result shows the MSD is always 0 in both of the previous method and the new method. Both method can locate feature point pairs precisely and estimate the translations without error.

Next, we use our new method with IEA to estimate the 3D rotations in x-, y- and z-axis. All the rotation angles are positive integers in [0, 5]. Fig. 6 shows the MSD for 3D rotations. Result shows that the new method with the IEA has smaller MSD.

We also applied our new method with IEA to estimate the combinations of translations and rotations with some random noises (Figs. 7-10). All the translations and rotation angles are positive integers in [0, 5]. The random noise is in the range of [0%, 5%]. Results show that the new method with IEA has smaller MSD.

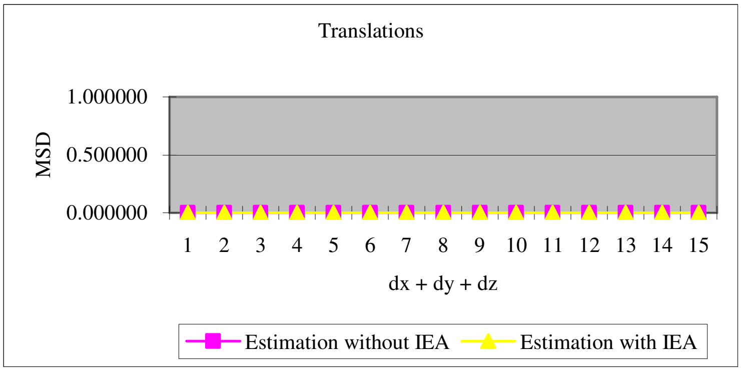


Fig. 5. MSD for 3D translations

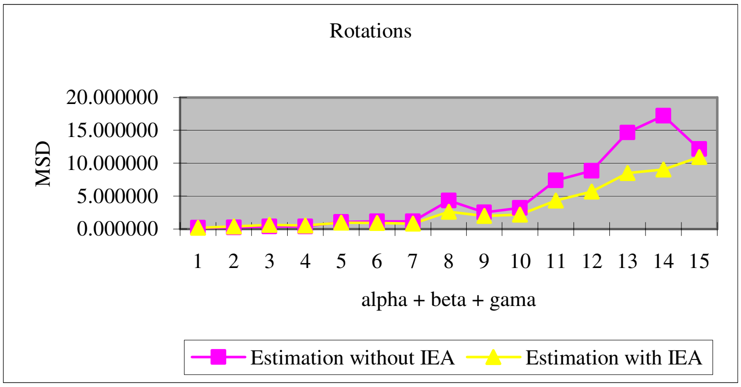


Fig. 6. MSD for 3D rotations

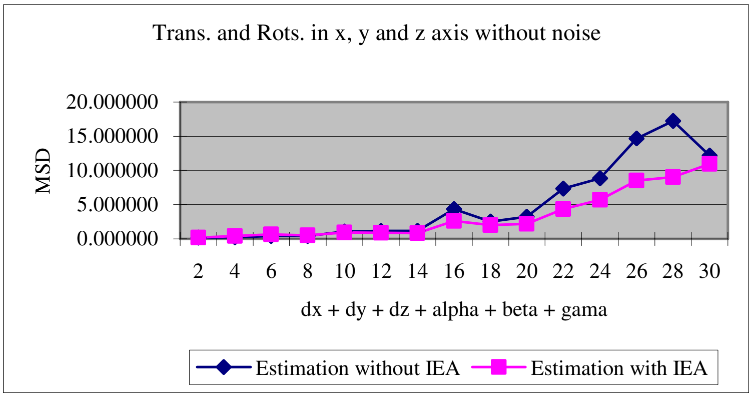


Fig. 7. MSD for 3D translations and rotations without noise

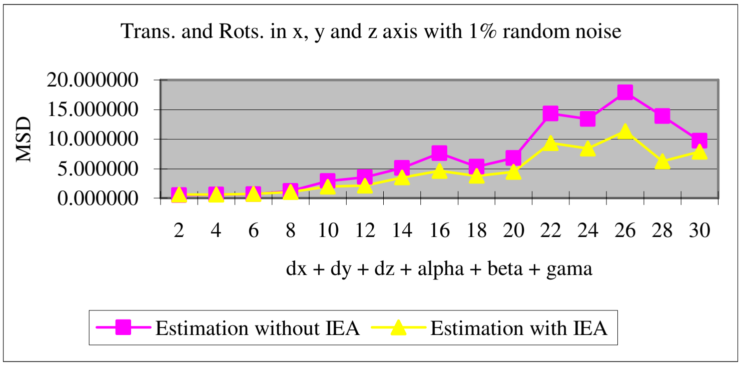


Fig. 8. MSD for 3D translations and rotations with 1% random noise

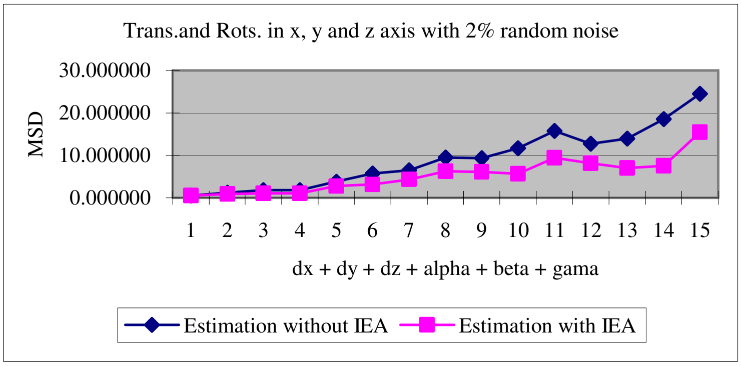


Fig. 9. MSD for 3D translations and rotations with 2% random noise

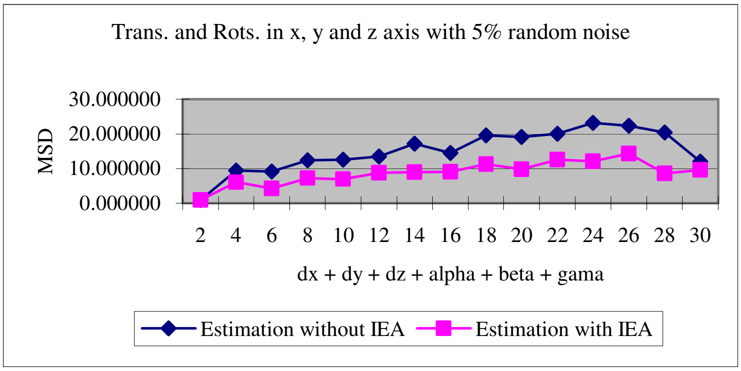


Fig. 10. MSD for 3D translations and rotations with 5% random noise

7 Conclusions and Future Work

In this paper, we demonstrated how to use an iterative algorithm to improve the estimation accuracy of 3D transformations. Result shows that this method works quite well when the differences in orientation are small. This work is a preliminary approach for defining and tracking the volume changes of airways with surgery. Because of constraints on time, we do not have airway model after surgery. Therefore, we had to create some models artificially to test our algorithm. In the near future, we will obtain real models after surgery and then compare our results using data from before and after surgery.

Acknowledgements

The support of ASRA and NSERC in making this work possible is gratefully acknowledged.

References

1. J.B.A. Maintz and M.A. Viergever, A Survey of Medical Image Registration, Medical Image Analysis, vol.2, p.1-36, 1998.
2. L. G. Brown, A survey of image registration techniques, ACM Computing Surveys (CSUR), 24(4), p.325-376, 1992.
3. P. J. Besl and N. D. Mckay, A method for registration of 3D shapes, IEEE Trans. Pattern Anal. Machine Intell. 14(2), p. 239-256, 1992.
4. João Maciel, Global matching: optimal solution to correspondence problems, PhD Thesis, Universidade Técnica de Lisboa, 2002
5. http://www.sph.sc.edu/comd/rorden/dicom.html
6. http://www.simpleware.com/software.php
7. D. Hearn and M. Baker, Computer Graphics, Pearson Education.
8. C. M. Ma and M. Sonka, A fully parallel 3D thinning algorithm and its applications, CVIU, 64 (3), p. 420-433, 1996
9. A. Chaturvedi, Z. Lee, Three-dimensional segmentation and skeletonization to build an airway tree data structure for small animals, 50(7), Physics in Medicine and Biology, p. 1405-1419, 2005.
10. J. J. Leader, Numerical analysis and scientific computation, Pearson Addison Wesley, Boston, 2004.
11. P. Torr. Motion Segmentation and Outlier Detection. PhD thesis, U. Oxford, 1995.
12. Justin Domke and Yiannis Aloimonos, A Probabilistic Notion of Correspondence and the Epipolar Constraint, Third International Symposium on 3D Data Processing, Visualization and Transmission, June 2006.
13. Wei Zhang and Jana Kosecka, Image based Localization in Urban Environments, Third International Symposium on 3D Data Processing, Visualization and Transmission, June 2006.

Temporal Alignment of Time Varying MRI Datasets for High Resolution Medical Visualization

Meghna Singh, Anup Basu, and Mrinal Mandal

Departments of Electrical Engineering and Computing Science, University of Alberta
meghna@ece.ualberta.ca, anup@cs.ualberta.ca,
mandal@ece.ualberta.ca

Abstract. Four-dimensional (4D) visualization of medical data, which entails the addition of time as the fourth dimension to 3D data, is fast gaining ground as a tool for diagnosis and surgical planning by medical practitioners. However, current medical image acquisition techniques do not support high-resolution 4D capture. Instead, multiple 3D datasets are acquired and a temporal relation is computed between these datasets in order to align them in time. In past work we presented a method of temporal alignment of MRI datasets to generate high-resolution medical data, which can be extended to 4D visualization. In this work, we present the details of our temporal alignment algorithm and also present comparative analysis in order to highlight the advantages of our method.

1 Introduction

The ability to create 4D data from MRI offers many advantages for visualization for medical practitioners, including (1) not subjecting patients to harmful radiation from X-rays, (2) short inter-exam time interval and most importantly (3) being able to view soft tissue structure in motion as the body functions to sustain life. Current technology allows users to view 3D data. However, the temporal changes in the 3D volumes are lost as no technique has 4D acquisition capability. In order to obtain high temporal resolution, accuracy in the spatial domain has to be sacrificed, as there is a trade-off between spatial and temporal resolution in MRI. Acquiring high spatial resolution data takes time, and within that period of acquisition, the anatomical structures in the body have already undergone motion, which leads to poor temporal resolution.

Our approach to addressing the 4D visualization problem is to use event dynamics to combine multiple datasets. So, while we can only capture a few representative images of the event because of limitations on acquisition speed, we can capture multiple repetitions of the event and thus more information about the event. Then, using the event dynamics as a guide, we can align these multiple datasets, which were acquired independently. The proposed method can find application in other areas such as spatio-temporal interpolation of video sequences, video frame rate up-conversion, multi-view visualization, super-resolution video generation and time series analysis of genomic/proteomic data. We define temporal alignment (or temporal registration) as the establishment of a correspondence or a temporal relation between two (or more) sequences or motion patterns, with a monotonic alignment, but without the necessity that the image pairs or sequences occur at the same point in time. Fig.1 shows two continuous trajectories, which are discretely sampled at time instances marked by the solid markers (dots and stars on the curves). Fig.1 (a) shows an integer alignment where a strict correspondence is found between each time sample. Fig.1 (b) shows a sub-frame alignment, where a non-integer correspondence is found between each time sample.

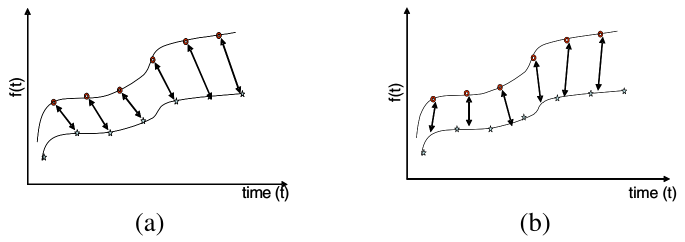


Fig. 1. Temporal registration/alignment of two trajectories. (a) Integer frame alignment, (b) Sub-frame alignment.

Currently, most temporal registration algorithms in medical imaging require the use of an external timestamp, such as an ECG signal from the heart (cardiac imaging)[8] or speech patterns recorded as audio data (tongue movement) [3] for computing the correspondence between datasets. Others, such as Perperidis [4][5], use changes in volume of the heart as timestamp information. Cardiac imaging has the advantage of possessing a periodic, repeating temporal scale. Other physiological functions, such as swallowing, do not possess a periodic, repeatable nature. Thus, multiple swallows can differ from each other in terms of the time duration of each of the three stages of a swallow. In our previous work [1], we presented preliminary results of fitted curve matching to generate high temporal resolution MRI video of swallowing. In [1], we showed that the dynamic properties of the water being ingested during a swallow could be used as a suitable timestamp to align multiple swallow datasets together. In this paper, we provide a comparative study of temporal alignment using fitted event dynamics [1] and a standard algorithm proposed by Caspi [6], which has been used by many researchers in the past.

This rest of this paper is organized as follows. In Section 2 we discuss some contemporary works related to temporal registration and alignment of sequences. In Section 3 we discuss in detail the temporal alignment algorithm proposed by Caspi [6] and present a case study where the Caspi algorithm will give erroneous results; we then present our fitted curve-matching algorithm. In Section 4 we present our experimental setup and results. In Section 5 we put forward the conclusion of our study and present some ideas for future work.

2 Review

In this section, we will review some past work in temporal alignment and registration of sequences. The reviewed papers deal with both space and time alignment, and model the registration as a two part transformation comprising of separable temporal and spatial transformations. Since our interest is in temporal alignment we only review the temporal alignment strategies of contemporary work.

Listgarten *et al.* [9] use a Hidden Markov Model (HMM) based approach for alignment of continuous time series data from speech and mass spectrometry. They present a continuous profile model (CPM) (much like Profile HMMs) which assumes that each acquired or observed time series is a noisy sub-sampled representation of a single true time series, which they call latent trace. The noisy time series are generated by moving through a sequence of Hidden Markov states. The CPM is trained using expectation maximization (Baum-Welch algorithm), and subsequently the latent trace of the model which represents a higher resolution series is obtained. However, the CPM algorithm only performs global alignments and a large number of replicated experimental data is required to train the HMM. Other constraints are that the nature of emission probabilities (Gaussian, in the case of [9]) and penalty term are dependent on the experimental data.

Giese and Poggio [7] model biological motion patterns using linear combinations of prototypical sequences with the objective being to recognize and synthesize images of objects. They concede that the computation of space-time correspondences between image sequences is a central problem and they use a dynamic time warping approach to account for the differences in time duration of activities. Other such as [10] have also used DTW for alignment of curves. [7] model the temporal deformation as a non-parametric transformation τ as shown in Eq. 1 below.

$$t' = t + \tau(t) \tag{1}$$

However, the underlying problem of correspondence and warping is ill-posed. As, even in the absence of ambiguity in the features being compared, there are infinitely many possible solutions that can bring the two trajectories into correspondence. The correspondence algorithm Giese *et al.* have developed has two stages. In the first stage they use dynamic time warping to solve a discrete optimization problem; in the second stage they solve for quasi-continuous spatial and temporal shifts by solving an optimization problem derived from linear interpolation between keyframes. They test their algorithm on human gait sequences, where the features are manually selected, and on synthetic motion patterns generated from an animated stick figure. However, in medical images it is often difficult to choose features manually (or via computer vision) and error in feature selection will propagate through the algorithm to give erroneous alignment.

Caspi *et al.* [6][2], approach the subject of spatio-temporal alignment of sequences by modeling the temporal offset between two image sequences as a 1D affine transform, which is computed as a translation that minimizes an error function. We discuss their approach in more detail in Section 3, and from hereon we will refer to their work as the *Caspi model.* Their 1D affine transform can be expressed by the following equation:

$$t' = s.t + \Delta t \tag{2}$$

Where 's' is the ratio between the frame rates of the two cameras and Δt is a sub-frame displacement, which is to be computed. They deal with sequences of events that are captured at the same time with a spatial and/or temporal misalignment. Thus the multiple sequences are of a single event. So while the temporal patterns can be offset from each other with a subframe displacement, they are subsampled patterns from the same real-life trajectory. On the other hand, the problem we address is to align temporal sequences, which were taken from two different repetitions of the same event, hence, we have two trajectories from two separate instances of an event.

Perperidis *et al.* [4], use the Caspi model [6] for temporal registration of cardiac images and enhance it by incorporating local deformable transformations using 1D cubic B-splines. Their temporal transform is represented as follows:

$$T_{temporal}(t) = T_{temporal}^{global}(t) + T_{temporal}^{local}(t) \tag{3}$$

The global transform has been modeled like Caspi's temporal transformation model, as shown in Eq 4, where α accounts for scaling differences and β accounts for translation differences.

$$T_{temporal}^{global}(t) = \alpha t + \beta \tag{4}$$

The local transform has been modeled as splines using the following equation (for more details refer to [5]):

$$T_{temporal}^{local}(t) = \sum_{l=0}^{3} B_l(u)\varphi_{t_{i+1}} \tag{5}$$

The optimal temporal transform is found by maximizing the normalized mutual information between the cardiac datasets. The local temporal transform is also computed separately by finding transitional landmarks and then searching through all possible local deformations of the splines while optimizing the normalized mutual information. They report that computing the optimal deformation by their approach is computationally expensive and sometimes takes over 24 hours to resolve. Their spline model is based on computing transitional landmarks such as start of the cardiac cycle, maximum contraction, end diastole etc., and depends greatly on the accuracy of this landmark recovery stage. For trajectories where such local landmarks are not as easily distinguishable, the temporal transformation will reduce to a global transformation, and thus to Caspi's model.

It is for this reason that in this paper we will compare our work with Caspi's model and highlight cases where their model will fail and a global fitted curve matching method will give optimal results. Sub-frame alignment is greatly improved in cases where a good parametric fit can be found for fitted curve matching, rather than linear interpolation.

3 Comparative Analysis

In this section, we discuss the Caspi model in detail and present an intuitive case study where the model will fail. We then present our curve-fitted matching algorithm for comparison.

3.1 Caspi Model for Temporal Alignment

Caspi *et al.* [6] define temporal misalignment to occur when two input sequences have a time-shift or offset between them, which could have been caused by different frame rates of the cameras or delay in activating the cameras. They model this by a 1D affine transform as shown in Eq. 2. The optimum temporal alignment is computed by minimizing the following error function (Note that the equations from [6] have been adapted to reflect only a temporal misalignment).

$$\vec{P} = \arg\min_{\Delta t} \sum_{Trajectories} \left(\sum_{t \in Trajectory} \left\| p'(s.t + \Delta t) - p(t) \right\|^2 \right) \tag{6}$$

$p(t) = [x(t), y(t), t]$ is the spatial position of the feature point along the trajectory at point '*t*' in time. $p'(s.t + \Delta t)$ is the location of the corresponding feature point in the second sequence at time $t' = s.t + \Delta t$. Since t' is modeled as a subframe displacement, they linearly interpolate coordinate values at t' from the corresponding integer location $t_1 = \lfloor t' \rfloor$ and $t_2 = \lceil t' \rceil$. Error minimization is performed by computing Δt for the best linear interpolation value. The minimization is stopped when the residual error stops changing or when a given number of iterations are exceeded. They search for $\alpha = t' - t_1$ that minimizes the following equation (see Fig. 2 for a pictorial representation):

$$\min_{\alpha} \sum_{t} \left\| p'(t_1).(1-\alpha) + p'(t_2).\alpha - p(t) \right\| : \alpha \in [0...1] \tag{7}$$

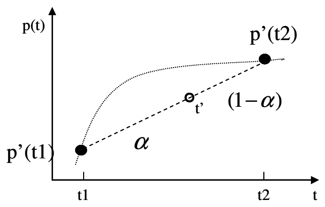


Fig. 2. Pictorial representation of linear interpolation in Eq.7 for sub-frame temporal alignment

Let us now consider a case of two trajectories $p(t)$ and $p'(t)$ as shown in Fig. 3, where $p'(t)$ is the same as trajectory $p(t)$, but offset by a subframe displacement '*true* Δt.' These continuous trajectories are discretely sampled by an acquisition source at different points in time. We represent this by the circles on the trajectory. Using Caspi's algorithm, $p'(t)$ will be interpolated for all subframe values from its value at the integer points, say $p'(t_1)$ and $p'(t_2)$ according to Eq 7. Subsequent to linear interpolation, at some temporal displacement Δt, $p'(t + \Delta t)$ will become equal to $p(t)$ and the algorithm will find a best match and stop (we have simplified the trajectory to that of a single coordinate, and only display two points on the trajectory). However, the true temporal displacement was '*true* Δt', which was not computed as the linear interpolation ignored the curvature of the trajectory. In cases where the sampling rate of the trajectory is high, linear interpolation will provide acceptable results. This is because, as the time been two samples begins to decrease, the trajectory becomes more linear. However, for poor temporal sampling rates, as is the case with MRI, linear interpolation will often result in erroneous temporal offsets, as we prove in our experiments with synthetic data in Section 4.2.

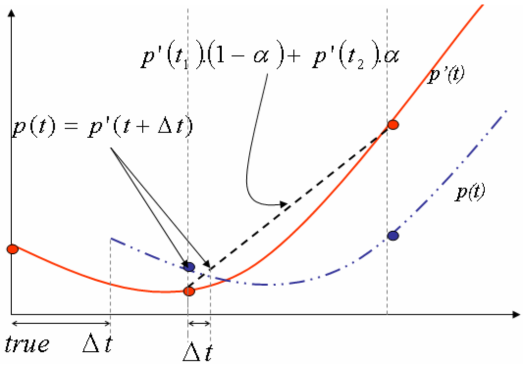


Fig. 3. Case study of failure of linear interpolation for temporal alignment

3.2 Curve Fitted Temporal Alignment

In our method, we do not consider individual trajectory points, but rather look at the entire dynamics that those points represent for alignment. We find an optimum least squares parametric fit which can be polynomial, exponential or linear. The fit that results in the least residual and best goodness of fit statistics is chosen to represent the trajectory. We have chosen a weighted (bi-square weights) least squares algorithm for our parametric fitting. The weighted least squares regression minimizes the error estimate in Eq. 8, where c_i are the trajectory points available, $\hat{c}_i$ are the trajectory points computed based on the least squares fit and w_i are the weights assigned to each trajectory point.

$$SSE=\sum_{i=1}^{n} w_i\left(c_i-\hat{c}_i\right)^2 \tag{8}$$

In the bi-square weighted method, the weight given to each data point depends on how far the point is from the fitted line. Points near the line get full weight while points farther from the line get a reduced weight. This approach minimizes the effect of outliers on the results of the curve fitting. The degree of the polynomial fit is decided by computing three goodness of fit statistics: sum of squared error (SSE), R-square and root mean square error (RMSE).

The SSE measure is defined in Eq. 8; a value closer to 0 indicates a better fit. The R-square statistic measures how successful the fit is in explaining the variation in the data and it is the square of the correlation between the trajectory point and the predicted trajectory point. For SST, sum of squares about the mean $\overline{c}$, is defined as:

$$SST = \sum_{i=1}^{n} w_i (c_i - \bar{c})^2 \tag{9}$$

Then, the R-square statistic can be defined as:

$$R\_square = 1 - \frac{SSE}{SST} \tag{10}$$

An R-square value closer to 1 indicates that a greater proportion of the variation in the data has been accounted for. The RMSE statistic estimates the standard deviation of the randomness or error in the data. An RMSE value closer to zero indicates a better fit. If '*n*' is the number of trajectory points acquired and '*m*' is the number of fitted coefficients estimated from *n*, then the residual degrees of freedom are defined as $\upsilon = n - m$. The root mean square error is then defined as:

$$RMSE = \sqrt{SSE/\upsilon} \tag{11}$$

We did not use non-parametric fitting with interpolants and splines as they depend greatly on the accuracy of the control points. Since our trajectories are from multiple swallows, we want to reduce the dependency of the algorithm on local points or features, which may or may not have been accurately obtained from the multiple trajectories. We believe that the overall trend of the event will be a better global representation of the trajectory.

Let the discrete points sampled on the two trajectories be represented as $c(x, y, n)$ and $c'(x', y', n')$. Since we are dealing with temporal alignment only, we assume that the trajectories are spatially aligned and can therefore be represented as $c(x, y, n)$ and $c'(x, y, n')$. Subsequent to curve fitting by minimizing the residual error and optimizing the goodness of fit statistics, the trajectories can be represented as continuous curves $c(x,y,t)$ and $c'(x,y,t')$, where $t' = t + \Delta t$. Temporal registration of the two paths involves finding a sub-frame displacement Δt that minimizes the distance between the coordinate positions in the two *continuous* paths as follows:

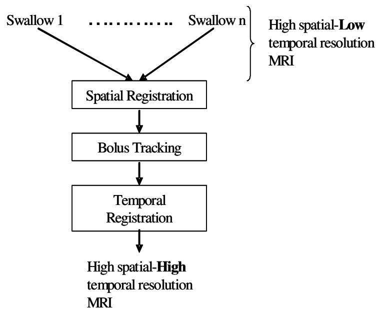
$$C = {}^{argmin}_{\Delta t} \sum_{paths} ((c_x - c_x^{'})^2 + (c_y - c_y^{'})^2) \tag{12}$$

4 Experimental Analysis and Results

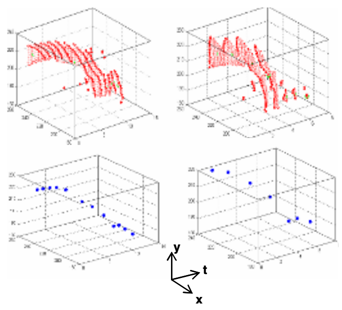
We divide our experimental analysis into two sections: Real MRI data and synthetic trajectories. Synthetic trajectory information allows us to compare the two techniques discussed above with respect to the actual truth; this truth information is unavailable for the real data.

4.1 Real Data

Our real data consists of two mid-saggital image sequences of a person drinking small amounts of water when lying inside an MRI scanner. For details on the acquisition and preprocessing please refer to [1]. An overview of the method is presented in Fig. 4 for the sake of completeness. The progression of water down the oral-pharyngeal tract is segmented and centroid coordinates are computed. The motion of the centroid is the trajectory path in the real data case. A polynomial of degree 6 was fitted on the coordinates of centroids with a residual error of 6.747, an R-square value of 0.9982 and a RMSE value of 1.29.


Swallow 1
....
Swallow n
High spatial-**Low** temporal resolution MRI
Spatial Registration
Bolus Tracking
Temporal Registration
High spatial-**High** temporal resolution MRI

(a)



(b)

Fig. 4. (a) Overview of curve fitted temporal alignment method. (b) Bolus path segmented from the MRI images and centroidal trajectories of two sequences.

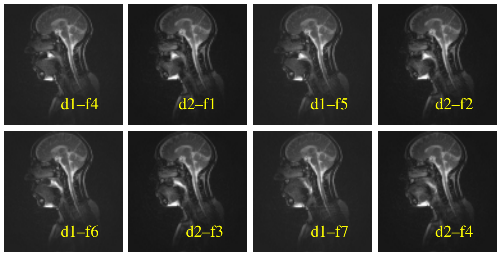


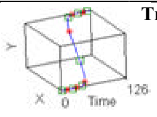
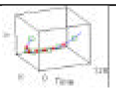
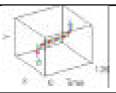
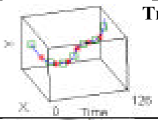
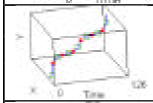
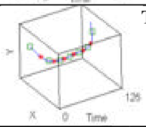
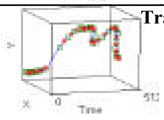
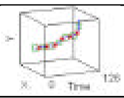
Fig. 5. Two MRI datasets of swallowing aligned using offset determined by fitted curve matching. Legend: d <sequence number>-f <frame number>. Based on centroidal paths of the bolus of water, frames from sequence 2 were placed approximately midway between frames from sequence 1, but offset from the beginning of sequence 1 by 3 frames for Caspi algorithm (not shown here) and offset=4 frames for curve fitted alignment model.

The results of temporal alignment with the Caspi model and the curve fitted alignment model were comparable. Caspi model found the alignment to be offset by 3.55 frames, while our method determined the offset to be 4.45. The temporal offsets were compared to each other by means of visual inspection (see Fig. 5) and since the ground truth information is unavailable, we cannot conclusively say that one method outperformed the other. However, the difference in performance is highlighted in the synthetic trajectory cases, where the ground truth information is available. We plan to conduct perceptual studies in order to evaluate the performance of the two algorithms for the MRI data.

4.2 Synthetic Data

Synthetic data was created by downsampling eight high temporal resolution trajectories into two subsampled trajectories each. One of these two downsampled trajectories was also offset by a known temporal difference in order to create temporal misalignments. Hence, for these cases the true temporal alignment was known *a priori*. Entire trajectories as well as sub-segments of trajectories were matched using both the models. Samples of the trajectories as well as the results of the temporal alignment are shown in Table 1. It can be seen from Table 1, that the curve fitted method has an average error of 0.86 frames, compared to 4.7 frames for the Caspi model. The maximum error for the Caspi model was 9.25 frames, while the error in our model was 0.75 frames for the same trajectory. The maximum error in our model was 4 frames, while the corresponding error for the Caspi model was twice as high at 8 frames.

Table 1. Results of temporal alignment with synthetic data

	Temporal alignment by linear interpolat-ion	**Temporal alignment by fitted curve matching**	**True temporal alignment**	**Error with linear Interpol-ation**	**Error with fitted curve matching**
 **Traj 1**	6	12.25	13	7	0.75
	4.75	12.25	13	8.25	0.75
	22.25	12.25	13	9.25	0.75
	7.25	13.5	13	5.75	0.5
 **Traj 2**	6	13.5	13	7	0.5
	6	13.5	13	7	0.5
	7.25	13.5	13	5.75	0.5
 **Traj 3**	6	16	13	7	3
	6	14.75	13	7	1.75
	12	11	10	2	1
 **Traj 4**	6	11	10	4	1
	12	10	10	2	0
	6	10	10	4	0
	12	11	10	2	1
 **Traj 5**	12	11	10	2	1
	6	10	10	4	0
	6	10	10	4	0
 **Traj 6**	6	10	10	4	0
	6	10	10	4	0
	6	9	10	4	1
	6	10	10	4	0
 **Traj 7**	6	11	10	4	1
	6	11	10	4	1
	6	11	10	4	1
	13	12	10	3	2
 **Traj 8**	12	10	10	2	0
	13	11	10	3	1
	18	14	10	8	4
Average error in frames				**4.7142**	**0.8571**

5 Conclusion and Future Work

In this paper we presented a comparative analysis of fitted curve matching and linear interpolation for sub-frame temporal alignment of MRI sequences and synthetic trajectories. The merit of the proposed method is that for fast occurring events with low acquisition rates, a fitted curve matching will compute temporal offsets to a higher accuracy than matching linearly interpolated values. It is to be noted that for such fitted curve matching, no landmark points need to be calculated on the trajectory and the computation time while larger than linear interpolation is still acceptably small: 0.091 seconds to compute an optimal polynomial fit and sequence matching for an 11 point trajectory using Matlab on an AMD Opteron270 2.0 GHz. Computation time for linear interpolation with the same input and settings was 0.012 seconds. In the future, we plan to analyze the effect of noise on the accuracy of the curve-fitted temporal alignment algorithm. Also, more experiments with medical data will be conducted in order to strengthen our claim.

Acknowledgment

The support of ASRA and NSERC in funding this work is gratefully acknowledged, as is the help of Richard Thompson and Carol Boliek in MRI acquisition.

References

1. Singh, M., Thompson, T., Basu, A., Rieger, J., Mandal, M.: Image Based Temporal Registration of MRI data for Medical Visualization, to appear in the IEEE Intl. Conf. on Image Processing ICIP, Atlanta, Georgia, Oct 2006.
2. Shechtman, E., Caspi, Y., Irani, M.: Increasing Space-Time Resolution in Video, European Conference on Computer Vision (ECCV), May 2002.
3. Yang, C., Stone, M.: Dynamic Programming Method for Temporal Registration of Three-dimensional Tongue Surface Motion from Multiple Utterances, Speech Communication, vol. 38, no. 1-2, pp. 201–209, 2002.
4. Perperidis, D., Mohiaddin, R., Rueckert, D.: Spatio-Temporal Free-Form Registration of Cardiac MR Image Sequences, MICCAI (1) 2004: 911-919.
5. Perperidis, D., Mohiaddin, R., Rueckert, D.: Spatio-Temporal Free-Form Registration of Cardiac MR Image Sequences, Medical Image Analysis, 9, pp 441-456, 2005.
6. Caspi, Y., Irani, M.: Spatiotemporal Alignment of Sequences, IEEE Trans on PAMI, Vol24, No.11, Nov 2002.
7. Giese, M.A., Poggio, T: Morphable Models for the Analysis and Synthesis of Complex Motion Patterns, Journal of Computer Vision, 38(1), pp 59-73, 2001.
8. Achenbach, S., *et al.*, Noninvasive Coronary Angiography by Retrospectively ECG-Gated Multislice Spiral CT, Circulation, Dec 2000; 102: 2823 - 2828.
9. Listgarten, S. R. J., Neal, R., Emili, A.: Multiple alignment of continuous time series, Advances in Neural Information Processing Systems 17. Cambridge, MA: MIT Press, 2005.
10. Wang, K., Gasser, T.: Alignment of curves by dynamic time warping, Annals Statistics, vol. 25, no. 3, pp. 1251–1276, 1997.

Physically Interacting with Four Dimensions

Hui Zhang and Andrew J. Hanson

Computer Science Department, Indiana University
huizhang@cs.indiana.edu, hanson@cs.indiana.edu

Abstract. We exploit the combination of a virtual world containing physically-interacting 4D objects with a multimodal haptics-driven user-interface model; the goal is to facilitate the development of accurate cognitive models enabling the visualization of 4D space. Our primary test domain supports tactile interaction with physically colliding and deformable curves and surfaces embedded in 4D, an important and challenging subject area of classical topology. We implement intricate interactions involving 4D curves and surfaces by haptically manipulating 3D projections of these objects.

1 Introduction

In the seventh book of the *Republic*, Plato introduces the allegory of the Cave, whose residents can only perceive the outside world via shadows thrown upon the walls, and who thus have only limited knowledge of the objects in the world. Interactive computer systems in fact work almost exclusively with shadows, i.e., representations of our 3D world cast upon 2D graphics screens by mathematical projection and rendering algorithms. Graphics methods allow us to add features to these shadows such as shading and occlusion that create perceptions far richer than the bleak shadows of Plato's vision, and which we interpret in our mental models as being truly 3D, despite the fact that in truth their dimension is reduced. Interactive control systems familiar to us all, such as the 1D steering wheel controlling the 2D motion of a car, or a 2D joystick controlling the 3D motion of an aircraft, provide further examples of physically-reactive controls that have lower dimensions than the domain of the object motion being manipulated.

Our task in this paper is to show how one can fully exploit the concept of projections to lower dimensions and physically reactive projection-based controllers to transform the task of interacting with 4D objects to a new level of physical reality. We like to think of this method intuitively as working in a "shadow world," a term widely used in the 4D visualization literature, with clear interactive implications and an ancient context in Plato's philosophy, though we will avoid the term for the most part due to implicit confusion with conventional computer graphics terminology. We start from the fairly familiar idea that a 2D mouse can control 3D objects represented as rendered images projected on a 2D screen, and extend that idea to a haptic interface that is restricted to a plane, but still empowered to sketch, touch, and take control of 3D objects projected to the controller plane. We then extend this concept upward by one dimension, creating 3D projections of 4D objects, rendering them with shading and occlusions, and providing 3D haptic controls and force-feedback in the resulting 3D world to sketch and control a 4D world. In this way we are able to create a realistic interactive interface that embodies the physical realities of a simulated 4D mathematical world of curves and surfaces just as standard 3D physical modeling embodies the properties of such objects in the human world. Thus we can make a non-trivial computer interface that correctly implements the intuitive physical properties of classes of 4D geometric problems whose comprehension is extremely challenging for the unaided human intellect.

2 Previous Work

The idea of cross-dimensional understanding has developed in many directions since Plato described his Cave. Abbott's *Flatland* asked how two-dimensional creatures might attempt to understand three-dimensional space [1,2]. Banchoff's pioneering work suggested how 3D computer-based projections could be used to study 4D objects [3,4]. Other representative efforts include a variety of ways to render 4D objects (see, e.g., Noll [5], Hollasch [6], Banks [7], Roseman [8], and Egli, Petit, and Stewart [9]), and to extend lighting model techniques to 4D (see, e.g., [10,11,12]).

Our own previous efforts have suggested how haptic exploration of 4D objects can exploit topological continuity by ignoring illusory 3D surface intersections and focusing on the intrinsic 4D geometry [13]. Typically, visual methods for understanding higher dimensions employ a projection to 3D as a fundamental step; this helps the viewer to identify salient global features of the whole object, and provides structural continuity when rotating either the object's (rigid) 3D projection, or performing higher-dimensional rotations to change the 3D projection. Haptic exploration, being intrinsically limited to the physical world, also must project higher dimensional objects to lower dimensions for exploration; within this context, surfaces embedded in 4D can be projected to 3D and explored topologically without regard to 3D artifacts to reveal complex topological relationships and structure. What is needed now to go beyond this framework is an enhancement of the environment that allows 4D intuition-building construction, interaction, weight, and deformation, as well as exploration of the 4D shapes themselves. The problem addressed here is therefore: "How can we physically interact with the fourth dimension?"

3 Motivation

People learn about the everyday world by combining sensory modalities, and knowledge of shape comes from a combination of sight, touch, and exploration. With a 3D touch-based computer interface, a 3D projection of, e.g., a 4D surface or curve can be used to explore intrinsic 4D geometry. We therefore use extensions of 3D methods to help us manipulate and comprehend the complicated case of shapes with collisions and weights in a 4D simulated world using a touch-based multimodal paradigm.

2D Example. To explain the basic features of our approach, we begin with a family of images corresponding to a 2D projection of the neighborhoods of the crossing points of two 3D curve segments. This could in principle be a pair of 2D curve segments, as though drawn with a pen on 2D paper. If all we can see is the pen strokes or the projected shadow of the 3D crossing, we find the result in Figure 1(a), which is devoid of 3D information. This problem is typically overcome by employing the "crossing diagram" method illustrated in Figure 1(b); this corresponds essentially to a depth-buffered rendering with some embellishments to emphasize discontinuities in depth. By thickening the curve to give it geometric structure and shading, we can get the additional improvement shown in Figure 1(c). Now we can begin to see how to exploit a haptic probe: if the probe is constrained to a 2D plane, the user can still edit the planar projections of curves and use modifier keys to specify over and under crossings to create intertwined 3D objects back in the "real" 3D space. Hence, the 2D projection supports 3D interaction.

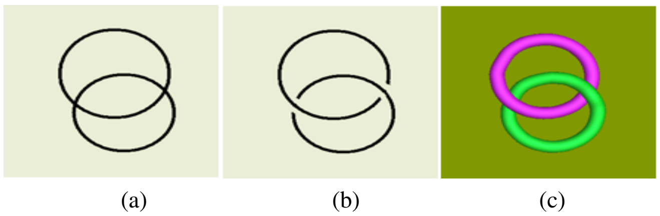


Fig. 1. (a) 2D projection diagram of crossing curves for two 3D rings. (b) Knot diagram representation of the linked rings provides 3D depth information. (c) Rendering with light and material adds apparent 3D geometry and shape to the 2D image.

3D Example. The technique we just used for projecting surfaces from 3D to 2D has an exact analog that applies to projection from 4D to 3D. We find that the metric properties are easier to understand if we use an orthogonal projection, but in selected circumstances, perspective 4D projection from the focal point of a 4D pinhole camera can also serve to reveal essential structures. The typical side-effect is that the resulting surfaces intersect in the 3D projection, even when in 4D space there are no intersections or singularities of any kind.

Figure 2 shows the 3D projection of a 4D ribbon linked with a 4D spherical surface, giving precise analogs to Figure 1, with the "bare shadow" in Figure 2(a), the crossing diagram in 2(b), and 4D depth pseudocoloring in 2(c,d).

Projection Editing. The key ideas of the overall scenario can now be summarized as follows:

- *Create a projection ("shadow") space in one lower dimension.*
- *Edit projected images of objects embedded in the higher dimension.* We must account for the lost depth information from the extra dimension, and cope with possibly having the haptic probe's status confused when one segment collides with another in the projection.
- *Adding an extra $\frac{1}{2}$ dimension.* Sketching in the reduced-dimension space can be supported by explicitly defining over and under crossings displaced slightly in the extra dimension when apparent collisions occur in the projected (shadow) image.
- *Create a crossing-diagram object.* This is basically the schematic equivalent of a 4D volume depth buffer.

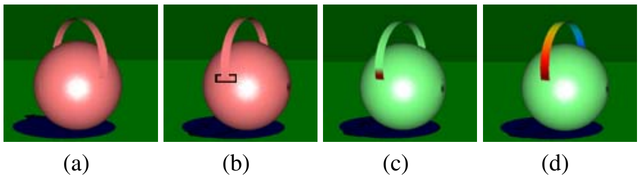


Fig. 2. (a) Two intersecting surfaces in 3D, typically resulting from the projection of a 4D scene to 3D. (b) Crossing diagram of the surfaces; the front portion of the ribbon surface is closer to the 4D projection point than the spherical surface. (c) Above-below crossing markings using 4D depth coding. (d) Color depth coding of ribbon modified by smoothing its 4D depth; just the top of ribbon is now at same 4D distance from the projection point as the entire sphere.

- *Enhance the geometry and depth information in the higher dimension.* Exploit computer graphics and visualization methods to enhance the perception of the geometry by color depth coding and exaggerated occlusion or crossing diagrams.
- *Physically interact with the higher dimensions.* By mapping the projected lower-dimensional (shadow) user input to the full-dimensional object along the projection rays, we can sense physical artifacts such as collision and gravity in the higher dimensional simulation.

4 Projection Editing Using Haptic Methods

We now describe the critical haptic portion of our interface. There have been a number of interesting attempts to exploit intuitive haptic interfaces to improve the sense of realism and to enhance the manipulation of virtual objects (see, e.g., [14], [15], and [16]). Another direction of research has focused on the haptic exploration of unknown objects by virtual fingers (see, e.g., [17] and [18]).

4.1 Design and Implementation

Procedure. The basic design of the system relies on the construction of a projection from 4D to 3D that can optionally annotate the under-over crossings in the projection, as well as supporting 6-degree-of-freedom 4D rotations to reposition the 3D projection arbitrarily to support the particular visualization requirements.

Constrained Haptic Space. Our basic force model simulates a “sticky” stylus in the projection space using a damped spring configuration model[19].

Collision Avoidance in Projected Space. When editing 2D projections of 3D curves or 3D images of surfaces embedded in 4D, collision detection and state management are essential issues. During editing, collision detection is continuously enabled to detect whether the object collides with itself or with other scene objects. The OpenHaptics HDAPI has been exploited in our implementation.

Making Over and Under Choices. When a collision occurs between a piece of an edited object and an existing object in projected space, users must make explicit over and under

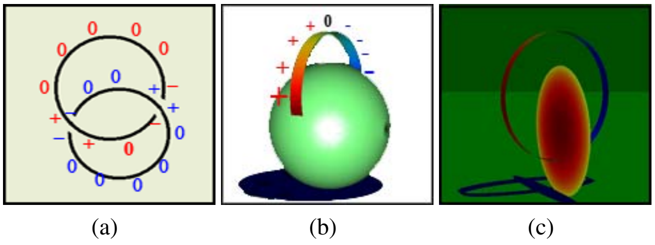


Fig. 3. (a) Recovering 3D depth from the 2D diagram.(b) Recovering 4D depth from the 3D diagram. (c) Projected image in YZW space with smoothed 4D depth.

choices before they can pass through the intersection. Thus the interface has a kind of interactive collision avoidance protocol.

4.2 Results

Editing Projections of 3D Curves. Traditional methods used by mathematicians for sketching knots and links make heavy use of crossing diagrams, and interactive graphics methods can assist this process (see, e.g., [20] and [21]). Using haptics-based projection-space editing can extend this paradigm still further. Figure 3(a) illustrates a simple example of the link-construction process; the resulting 3D curves in Figure 1(c) have been smoothed using the Minimum Distance Energy method [22] to fill out the $2\frac{1}{2}$D sketches to make a more natural shape.

Editing Projections of 4D Surfaces. We can analogously create 4D links working with the projected images of the 4D surfaces. We depend on the 3D collision mechanisms to locate possible crossings, and rotate or vary the 3D projection axes to help us see the 4D over and under crossings analogous to Figure 3(a), as shown in Figure 3(b). Applying a smoothing algorithm to the 4D depth of the sketched ribbon results in the more natural shapes of Figure 3(c).

5 Collision in Four Dimensions

Applying general rotations to objects projected from higher dimensions smoothly alters the projected images. For example, if projections of 3D rings falsely appear to be linked, a properly chosen 3D rotation will detach the two projected images from each other. The analogous observation holds when we apply 4D rotations to 4D objects represented by their 3D projections, as shown in Figure 4(a)-(c).

In this section, we introduce a set of methods for physically detecting, manipulating, and sensing collisions and physical forces in four dimensions to go beyond what can be done with crossing diagram methods or by applying general rotations alone.

5.1 4D Collision Detection

To understand the nonintuitive mechanisms of 4D collision, let us start with a pair of two-dimensional planes through the origin in four-dimensional space (see Figure

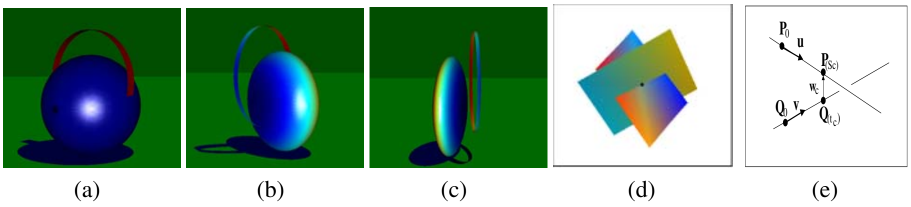


Fig. 4. (a) The projection of two unlinked 4D embedded objects can appear linked. (b) 4D rotation changes the projected images. (c) A particular 4D rotation detaches the 3D projected images.(d) 4D Collision: surfaces are color-coded to indicate their height in four-space relative to the projection point, so we observe that there is just one pair of points with the same fourth coordinate as well as the same first three coordinates. (e) Closest Points between 4D Lines.

4(d)(e)). The two squares intersect in a single 4D point at the origin. In the three-dimensional projection, although the planes appear to intersect along an entire line, when the surfaces are 4D depth color-coded, we can see that there is just one pair of points with the same fourth coordinates as well as the same coordinates in the 3D projection. *Note that 4D collisions take place only if there is also a 3D collision in the projection, but that 3D collisions in the projection may not imply 4D collisions.*

4D Collision Detection Based on Projection. Figure 4(e) illustrates the basic case for 4D Collision Detection; a 4D depth collision test must be performed along the intersecting lines of the projected images of 4D surfaces. 4D collision occurs if, and only if, one or more pairs of points have the same fourth coordinate along the intersecting line.

Now let 4D surfaces $\mathbf{S}_A$ and $\mathbf{S}_B$ intersect in the 3D projected image along the line segment $\mathbf{L}_{\mathrm{se}}$, from point $\mathbf{P}_s = (x_s, y_s, z_s)$ to point $\mathbf{P}_e = (x_e, y_e, z_e)$. Suppose the pair of 4D points sharing $\mathbf{P}_s$ as projected points are $\mathbf{P}_0$ on $\mathbf{S}_A$, and $\mathbf{Q}_0$ on $\mathbf{S}_B$; likewise, we have $\mathbf{P}_1$ on $\mathbf{S}_A$, and $\mathbf{Q}_1$ on $\mathbf{S}_B$ sharing $\mathbf{P}_e$. Obviously, $\mathbf{P}_0$, $\mathbf{P}_1$, $\mathbf{Q}_0$, and $\mathbf{Q}_1$ can be represented as:

$$\begin{aligned}
\mathbf{P}_0 &= (x_s, y_s, z_s, w_{sA}) \\
\mathbf{Q}_0 &= (x_s, y_s, z_s, w_{sB}) \\
\mathbf{P}_1 &= (x_e, y_e, z_e, w_{eA}) \\
\mathbf{Q}_1 &= (x_e, y_e, z_e, w_{eB})
\end{aligned} \tag{1}$$

Now we can detect the closest approach of the intersection lines contained in each plane, and prevent actual 4D collisions from occurring before they happen based on keeping the closest approach greater than some suitable small number τ.

Closest Points between 4D Line Segments. We first consider two infinite lines $\mathbf{L}_1$: $\mathbf{P}(s) = \mathbf{P}_0 + s(\mathbf{P}_1 - \mathbf{P}_0) = \mathbf{P}_0 + s\mathbf{u}$ and $\mathbf{L}_2$: $\mathbf{Q}(t) = \mathbf{Q}_0 + t(\mathbf{Q}_1 - \mathbf{Q}_0) = \mathbf{Q}_0 + t\mathbf{v}$. Let $\mathbf{w}(s,t) = \mathbf{P}(s) - \mathbf{Q}(t)$ be a vector between points on the two lines. We want to find the $\mathbf{w}(s,t)$ that has a minimum length for all s and t. In any N-dimensional space, the two lines $\mathbf{L}_1$ and $\mathbf{L}_2$ are closest at unique points $\mathbf{P}(s_c)$ and $\mathbf{Q}(t_c)$ for which $\mathbf{w}(s_c, t_c)$ attains its minimum length. Also, if $\mathbf{L}_1$ and $\mathbf{L}_2$ are not parallel, then the line segment $\mathbf{P}(s_c) \leftrightarrow \mathbf{Q}(t_c)$ joining the closest points is uniquely perpendicular to both lines at the same time. No other segment between $\mathbf{L}_1$ and $\mathbf{L}_2$ has this property(see Figure 4(e)). That is, the vector $\mathbf{w}_c = \mathbf{w}(s_c,t_c)$ is uniquely perpendicular to the line direction vectors $\mathbf{u}$ and $\mathbf{v}$, and thus it satisfies the equations:

$$\begin{aligned} \mathbf{u}\cdot\mathbf{w}_c &= 0 \\ \mathbf{v}\cdot\mathbf{w}_c &= 0\,. \end{aligned} \tag{2}$$

We can solve these two equations by substituting $\mathbf{w}_c = \mathbf{P}(s_c) - \mathbf{Q}(t_c) = \mathbf{w}_0 + s_c\mathbf{u} - t_c\mathbf{v}$, where $\mathbf{w}_0 = \mathbf{P}_0 - \mathbf{Q}_0$, into each one to get two simultaneous linear equations. Then, letting $a = \mathbf{u}\cdot\mathbf{u}$, $b = \mathbf{u}\cdot\mathbf{v}$, $d = \mathbf{u}\cdot\mathbf{w}_0$, and $e = \mathbf{v}\cdot\mathbf{w}_0$, we solve for s_c and t_c as:

$$s_c = \frac{be-cd}{ac-b^2}, \quad t_c = \frac{ae-bd}{ac-b^2}\,. \tag{3}$$

Having solved for s_c and t_c, we have the points $\mathbf{P}(s_c)$ and $\mathbf{Q}(t_c)$ where the two lines $\mathbf{L}_1$ and $\mathbf{L}_2$ are closest. Then the distance between them is given by:

$$d(\mathbf{L}_1,\mathbf{L}_2) = \left|(\mathbf{P}_0 - \mathbf{Q}_0) + \frac{(be-cd)\mathbf{u} - (ae-bd)\mathbf{v}}{ac-b^2}\right|\,. \tag{4}$$

Now we represent a segment $\mathbf{S}_1$ (between endpoints $\mathbf{P}_0$ and $\mathbf{P}_1$) as the points on $\mathbf{L}_1 : \mathbf{P}(s) = \mathbf{P}_0 + s(\mathbf{P}_1 - \mathbf{P}_0) = \mathbf{P}_0 + s\mathbf{u}$ with $0 \le s \le 1$. Similarly, the segment $\mathbf{S}_2$ on $\mathbf{L}_2$ from $\mathbf{Q}_0$ to $\mathbf{Q}_1$ is given by the points $\mathbf{Q}(t)$ with $0 \le t \le 1$. The distance between segment $\mathbf{S}_1$ and $\mathbf{S}_2$ may not be the same as the distance between their extended lines $\mathbf{L}_1$ and $\mathbf{L}_2$. The first step in computing a distance involving segments is to get the closest points for the lines they lie on. So, we first compute s_c and t_c for $\mathbf{L}_1$ and $\mathbf{L}_2$, and if these are in the range of the involved segment, then they are also the closest points for them. But if they lie outside the range, then they are not and we have to determine new points that minimize $\mathbf{W}(s,t) = \mathbf{P}(s) - \mathbf{Q}(t)$ over the ranges of interest.

To do this, we first note that minimizing the length of $\mathbf{w}$ is the same as minimizing $|\mathbf{w}|^2 = \mathbf{w}\cdot\mathbf{w} = (\mathbf{w_0} + s\mathbf{u} - t\mathbf{v})\cdot(\mathbf{w_0} + s\mathbf{u} - t\mathbf{v})$ which is a quadratic function of s and t. In fact, this expression defines a paraboloid over the (s,t)-plane with a minimum at $C = (s_c,t_c)$, and which is strictly increasing along rays in the (s,t)-plane that start from C and go in any direction. However, when segments are involved, we need the minimum over a subregion $\mathbf{G}$ of the (s,t)-plane, and the global minimum at C may lie outside of $\mathbf{G}$. An approach is given by [23], suggesting that the minimum always occurs on the boundary of $\mathbf{G}$, and in particular, on the part of $\mathbf{G}$'s boundary that is visible to C. Thus by testing all candidate boundaries, we can compute the closest points between 4D line segments.

Pair Reduction. In practice, interesting 4D surface models may consist of thousands of polygons, and manipulating these objects may require very costly searches to perform collision detection. A typical complexity-reduction strategy is to use a four-dimensional bounding box to eliminate pairs that have no 4D depth overlap at all. In addition, as noted earlier, although nearest approaches of 4D objects may not coincide in the 3D projection, any actual 4D collision must occur also in the 3D projection. This fact actually helps to reduce the collision detection problem to one lower dimension, and to accelerate the identification of starting points for nearest-approach computation.

5.2 4D Collision Management

Physics-based simulation of the 4D world involves challenging problems in 4D collision management. Self-collisions of flexible objects must be treated as well as collisions between distinct 4D objects, both rigid and flexible.

4D self-collision management methods follow exact parallels to 3D collision management (see, e.g., [20], [24]). The most basic form of self-collision treatment is as follows: When two 4D facets are detected to be at a distance $d < \tau$ from each other (i.e., the two corresponding facets are defined to be colliding if they are closer than a minimum distance τ), the pair of closest points in the facets is identified and Δ is defined as the 4D line passing through them. Then an equal (but opposite) displacement is applied to each facet along Δ. This displacement is just large enough to take the facets out of collision range, with a slight "safety margin" ε to allow for a sliding motion. Standard 3D self-collision mechanisms must also be extended to apply Newton's laws to four dimensions in a natural way. By stretching, compressing, bending, and twisting the 4D object while keeping its topological structure, we can convert the mathematical abstraction of any 4D object into an interactive reality.

4D collisions between distinct 4D rigid objects are handled in a similar way to self-collisions.

6 Physically Interacting with Four Dimensions Using Haptic Methods

Physics-based simulations of 4D collision detection and management are the key to creating a bridge to four-dimensional reality. 3D physics-based simulation has had broad success in many modeling domains, and the work closest to ours involves chains (see, e.g., [25]) and deformable objects (see, e.g., [26], [27],[20], and [28]). Haptic interfaces have also been used to implement realistic 3D physics-based simulations (see [14] and [16]). By extending these approaches to 4D physics-based haptic simulation, we can establish intuitive interactions exposing the true nature of higher dimensions, even though our controls are restricted to the physical world, which is the 3D domain of 4D projections.

Example 1: Lifting a Chain in Four Dimensions. A classic 4D structure is a "chain" consisting of linked spheres and circular ribbons. The forces to be considered include:

- Forces applied to a body by the haptic probe. The force is constrained to the projection domain in which it is applied, with the unseen projection-direction coordinate held fixed. Just as one can lift 2D projected images of 3D rings with control constrained to the 2D projection plane, we can lift 3D projected images of 4D object configurations in the same way.
- Force of gravity and damping. Allow objects to hang from one another realistically as they are lifted against the force of gravity by contact with one another.

The mechanism of 4D lifting of the chain can be implemented via the following cycles (see Figure 5):

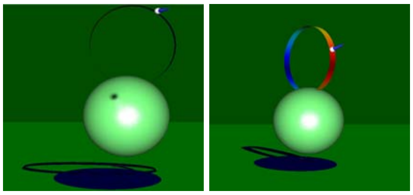


Fig. 5. The haptic interface for lifting a chain element in four dimensions, sensing collisions and weight

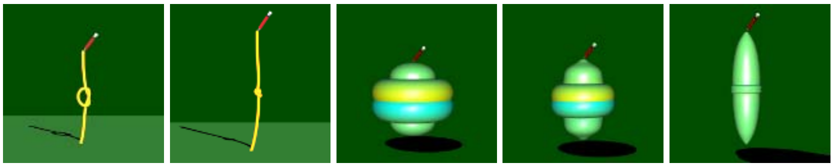


Fig. 6. Left: Tightening a 3D curve. Right: Tightening a 4D knotted sphere by pulling in 3D.

↪ Move the haptic proxy, attached to a ribbon at a user-specified location, in projection space.
↪ Continue until a potential collision is detected between the ribbon and an adjacent sphere.
↪ Locate closest points, prepare forces for application to 4D objects.
↪ Activate contact forces and gravity to adjust positions of objects.
↪ The haptic proxy is free again to move the ribbon with constrained motions.

Example 2: Tightening the Spun Trefoil Knot. Just as there are many ways to represent and alter the same 3D knot using equivalent crossing diagram representations, there are also many ways to represent the same knotted sphere as a projection from 4D to 3D. Manipulating these deformations via a 4D touch-sensitive interface in the projection to 3D allows one to verify the validity of the transformations among the various projected forms (corresponding typically to what a knot-theorist would call “Reidemeister moves.”).

In our version of the standard “spun-knot” construction [29] of a 4D knotted sphere, the user interactively sketches a 3D knot K using the 3D editing interface, then selects a “spin plane” to spin K into a 4-dimensional knot.

Tightening a 4D spun knot is similar to a 3D knot-tightening simulation. The FTL (Follow the Leader) algorithm described in [20] can be extended to the fourth dimension to compute the 4D knotted sphere tightening process (see Figure 6). The overall algorithm for tightening the spun trefoil knot is the following:

↪ Read the new position of the location grasped by the haptic proxy (typically, the north pole of the knotted sphere).
↪ Prepare and apply forces on the connected physical mesh components.
↪ Apply forces and compute the new knotted sphere configuration (global motion).
↪ Compute tentative (self-)collisions.

$\hookrightarrow$ If a potential collision is detected between two 4D segments or facets, apply collision forces with friction and gravity.
$\hookrightarrow$ The haptic proxy is free again to move the grasped location.

7 Conclusion and Future Work

Our implementation is based on a standard OpenGL graphics system with a high-performance graphics card, combined with SensAble Technology's Omni PHANToM force-feedback haptic device. Our user interface is based on OpenGL, SensAble's OpenHaptics toolkit, and a locally customized GLUI API.

We have created a multimodal computer interface to interact physically with the fourth dimension via its 3D projection. Our approach fluidly integrates visual information with haptic feedback and interaction. We have exploited these capabilities to build and interact with components of a 4D simulated world, a world that we can work with physically and that is no less real than a 3D computer simulation; we have thus presented a practical way to connect the mathematical abstraction of 4D objects to a visible, touchable interactive reality.

Among our objectives for future work are to extend the range of scene objects that we can support to include more complex knots, links, and Riemann surfaces, as well as three-manifolds in addition to curves and surfaces. In a more practical direction, one might imagine exploring high-dimensional information data sets by exploiting analogs of our interface acting along projection rays from the higher dimensional information space into a 3D control space.

References

1. Abbott, E.A.: Flatland. Dover Publications, Inc. (1952)
2. Dewdney, A.K.: The Planiverse: Computer Contact with a Two-Dimensional World. Poseidon Press (1984)
3. Banchoff, T.F.: Visualizing two-dimensional phenomena in four-dimensional space: A computer graphics approach. In Wegman, E., Priest, D., eds.: Statistical Image Processing and Computer Graphics. Marcel Dekker, Inc., New York (1986) 187–202
4. Banchoff, T.F.: Beyond the third dimension: Geometry, computer graphics, and higher dimensions. Scientific American Library (1990)
5. Noll, M.A.: A computer technique for displaying n-dimensional hyperobjects. Communications of the ACM **10** (1967) 469–473
6. Hollasch, S.: Four-space visualization of 4D objects. Master's thesis, Arizona State University (1991)
7. Banks, D.: Interactive display and manipulation of two-dimensional surfaces in four dimensional space. In: Symposium on Interactive 3D Graphics, ACM (1992) 197–207
8. Roseman, D.: Twisting and turning in four dimensions (1993) Video animation, Department of Mathematics, University of Iowa, and the Geometry Center.
9. Egli, R., Petit, C., Stewart, N.F.: Moving coordinate frames for representation and visualization in four dimensions. Computers and Graphics **20** (1996) 905–919
10. Carey, S.A., Burton, R.P., Campbell, D.M.: Shades of a higher dimension. Computer Graphics World (1987) 93–94

11. Steiner, K.V., Burton, R.P.: Hidden volumes: The 4th dimension. Computer Graphics World (1987) 71–74
12. Hanson, A.J., Heng, P.A.: Illuminating the fourth dimension. Computer Graphics and Applications **12** (1992) 54–62
13. Hanson, A.J., Zhang, H.: Multimodal exploration of the fourth dimension. In: Proceedings of IEEE Visualization. (2005) 263–270
14. Baxter, W.V., Scheib, V., Lin, M.C.: dAb: Interactive haptic painting with 3D virtual brushes. In Fiume, E., ed.: SIGGRAPH 2001, Computer Graphics Proceedings, ACM SIGGRAPH, ACM (2001) 461–468
15. Thompson, T., Nelson, D., Cohen, E., Hollerbach, J.: Maneuverable nurbs models within a haptic virtual environment (1997)
16. Kim, L., Sukhatme, G., Desbrun, M.: Haptic editing for decoration and material properties (2003)
17. Okamura, A.M.: Haptic Exploration of Unknown Objects. PhD thesis, Stanford University, Department of Mechanical Engineering, California, USA (2000)
18. Yu, W., Ramloll, R., Brewster, S.: Haptic graphs for blind computer users. In: First International Workshop on Haptic Human-Computer Interaction, British HCI Group, Springer (2000) 102–107
19. SensAble, Inc.: 3D Touch SDK OpenHaptics Toolkit Programmer's Guide. (2004)
20. Brown, J., Latombe, J.C., Montgomery, K.: Real-time knot-tying simulation. The Visual Computer **20** (2004) 165–179
21. Scharein, R.G.: Interactive Topological Drawing. PhD thesis, Department of Computer Science, The University of British Columbia (1998)
22. Huang, M., Grzeszczuk, R.P., Kauffman, L.H.: Untangling knots by stochastic energy optimization. In: VIS '96: Proceedings of the 7th conference on Visualization '96, Los Alamitos, CA, USA, IEEE Computer Society Press (1996) 279–ff.
23. Eberly, D.: 3D Game Engine Design. Morgan Kaufmann Publisher (2001)
24. Phillips, J., Ladd, A., Kavraki, L.: Simulated knot tying (2002)
25. Barzel, R., Barr, A.H.: A modeling system based on dynamic constraints. In: SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM Press (1988) 179–188
26. Terzopoulos, D., Platt, J., Barr, A., Fleischer, K.: Elastically deformable models. In: SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM Press (1987) 205–214
27. Terzopoulos, D., Witkin, A.: Physically based models with rigid and deformable components. IEEE Comput. Graph. Appl. **8** (1988) 41–51
28. Wejchert, J., Haumann, D.: Animation aerodynamics. In: SIGGRAPH '91: Proceedings of the 18th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM Press (1991) 19–22
29. Friedman, G.: Knot spinning. Handbook of knot theory (2005) 187–208

Low Level Moving-Feature Extraction Via Heat Flow Analogy

Cem Direkoğlu and Mark S. Nixon

Department of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ, UK
{cd05r, msn}@ecs.soton.ac.uk

Abstract. In this paper, an intelligent and automatic moving object edge detection algorithm is proposed, based on heat flow analogy. This algorithm starts with anisotropic heat diffusion in the spatial domain to remove noise and sharpen region boundaries for the purpose of obtaining high quality edge data. Then, isotropic heat diffusion is applied in the temporal domain to calculate the total amount of heat flow. The moving edges are represented as the total amount of heat flow out from the reference frame. The overall process is completed by non-maxima suppression and hysteresis thresholding to obtain binary moving edges. Evaluation results indicate that this approach has advantages in handling noise in the temporal domain because of the averaging inherent of isotropic heat flow. Results also show that this technique can detect moving edges in image sequences.

1 Introduction

The heat flow analogy has been deployed in various ways in image processing and computer vision. Five applications are briefly surveyed here: image smoothing and enhancement; region based image segmentation; thinning; active contours; and motion analysis.

Image smoothing and enhancement: Heat flow has first been used for image smoothing. Witkin [1] introduced scale-space theory which involves generating coarser resolution images by convolving the original image with a Gaussian kernel. Then Koenderink [2] and Hummel [3] pointed out that the family of derived images may be equivalently viewed as the solution of heat conduction or diffusion equation based on several criteria: causality, homogeneity and isotropy. According to homogeneity and isotropy, blurring is required to be spatially invariant which makes it difficult to obtain accurately the location of edges at coarse scales. Then, Perona and Malik [4] introduced anisotropic heat flow for selective image smoothing that avoids blurring and localization problems of the edges. In this process, the diffusion coefficient is allowed to vary according to the magnitude of the local image gradient. In this way, high quality edge detection is observed. After that, many approaches and models have been developed alternative to Perona and Malik's work. Some of them are: geometry driven heat flow [5], graph spectral model [6], probabilistic view [7], regularization method [8] and discrete image flux conduction model [9].

Region based image segmentation: In [10], the anisotropic diffusion pyramid (ADP) was introduced for region based segmentation. The pyramid is constructed using the scale space representation of anisotropic diffusion. Since anisotropic diffusion preserves edge locations as the scale increases, region boundaries in the coarse to fine ADP segmentation are accurately delineated. Recently, Manay and Yezzi [11] have proposed the anti-geometric heat flow model for adaptive thresholding and segmentation of regions. Here, anti-geometric heat flow is represented as diffusion through the normal direction of edges that smears rather than preserves them. As a result of this, regions on the opposite sides of prominent edges are captured in greyscale images.

Thinning: In [12], a new thinning algorithm was introduced based on time-reversed isotropic heat flow. Given an image, which is viewed as a thermal conductor, first the heat flow direction map is computed, then time-reversed heat conduction is simulated to get thinned a pattern. This algorithm can be applied to gray-scale or binary images.

Snake or Active Contours: Active contours can be classified as Parametric Active Contours and Geometric Active Contours (Level Sets). Parametric active contour (PAC) is the first snake model introduced by Kass et al. [13]. Problems associated with PAC are initialization and poor convergence to concave regions. These problems are largely solved with the development of new external force model which is called Gradient Vector Field (GVF) [14]. It is computed as diffusion of the gradient vectors of the grey level or binary edge map. This diffusion process arises from the heat conduction model. A geometric active contour [15, 16] is based on a curve moving in normal direction with its curvature dependent speed. This phenomenon is tackled with a level set method in higher dimension viewing the curve as the level set of zero. The curve movement on the level set is achieved with the geometric heat flow [5].

Motion analysis: Horn and Schunk [17] developed a method for optical flow (velocity vectors) computation from sequence of images. The concept includes two constraints which are change of brightness and smoothness of the velocity flow. An equation was developed whose progress is similar to the propagation effects in the solution of heat conduction equation. However, this method does not preserve optical flow discontinuities on the motion boundary because of the isotropy property of the smoothness constraint. Then some extensions have modified the smoothness constraint and observe anisotropic behaviour to preserve motion boundaries [18, 19]. Makrogiannis and Bourbakis [20] are the first who proposed spatio-temporal anisotropic heat diffusion for motion activity measurement. The motion activity measure is derived from the total amount of diffusion in the spatio-temporal domain. Then, this process is completed by kernel based density estimation and watershed-based segmentation of regions.

In this paper, a moving object edge detection algorithm is proposed based on heat flow analogy. This algorithm starts with anisotropic heat diffusion in the spatial domain to remove noise and sharpen region boundaries for the purpose of obtaining high quality edge maps. Once the enhanced edge maps are observed in consecutive frames, isotropic heat diffusion is applied in the temporal domain to calculate the total amount of heat flow. The moving edge map is represented as the total amount of heat flow out (-) from the reference frame. The overall process is completed by non-maxima suppression for thinning and then hysteresis thresholding to obtain binary moving edges. Evaluation results indicate that this approach has advantages in handling noise in the temporal domain because of the averaging inherent of isotropic heat flow. Results also show that this technique can detect moving edges in image sequences. The proposed algorithm is also shown in Fig. 1.

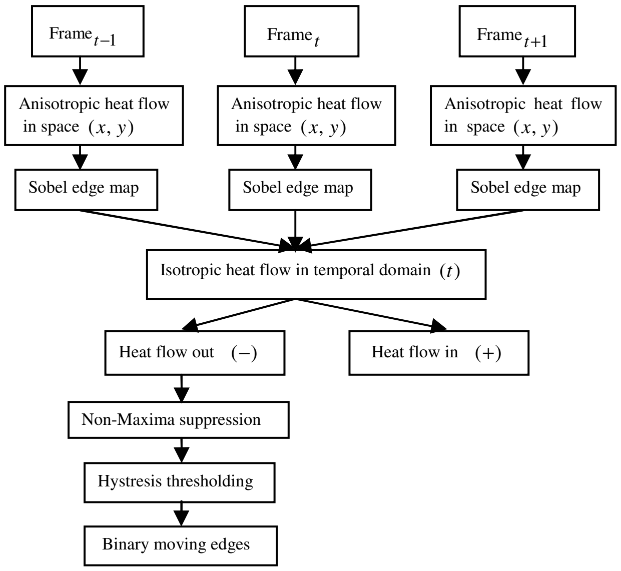


Fig. 1. Proposed algorithm for moving edge detection

This paper is organized as follows: Section 2 introduces the basic concepts of heat flow. Section 3 discusses the anisotropic heat flow for edge map enhancement. Section 4 introduces novel moving edge detection method. Section 5 concerns evaluation and experimental results, prior to conclusions.

2 Basic Concept of Heat Flow

Conduction, convection and radiation are three different modes of heat flow. Here, we chose to investigate use of a conduction model which operates well in our algorithm. Conduction is the flow of heat from the high temperature regions to the low temperature regions as time passes [21]. According to Fourier's law of heat conduction, the flow of heat per unit area and per unit time is,

$$F = -k\nabla T(x, y, t) \tag{1}$$

Where, F represents the heat-flow rate, k is a positive constant that is called the thermal conductivity of a material, $\nabla T(x, y, t)$ is the temperature gradient and the minus sign indicates that heat flows in the opposite direction of temperature gradient. If we consider a heat balance on a two dimension material as shown in Fig. 2,

$$\begin{tikzcd}\text{Heat flow in }(+) \arrow[r] & \boxed{T(x,y,t)+Q} \arrow[r] & \text{Heat flow out }(-)\end{tikzcd}$$

Fig. 2. 2D material for heat balance analysis

The change of temperature over time at each point of material is described using the general heat conduction or diffusion equation,

$$\frac{dT(x,y,t)}{dt} = \kappa \Delta T(x,y,t) + Q \tag{2}$$

Where, $dT(x,y,t)/dt$ is the rate of change of temperature, $\kappa \in [0,0.25]$ is the thermal diffusivity, Δ is Laplacian operator, $\kappa \Delta T(x,y,t)$ is total amount of heat flow in/out, and Q is a heat source. Note that in our application, the heat source is actually ignored because it only adds bias to the result.

3 Anisotropic Heat Diffusion and Edge Enhancement

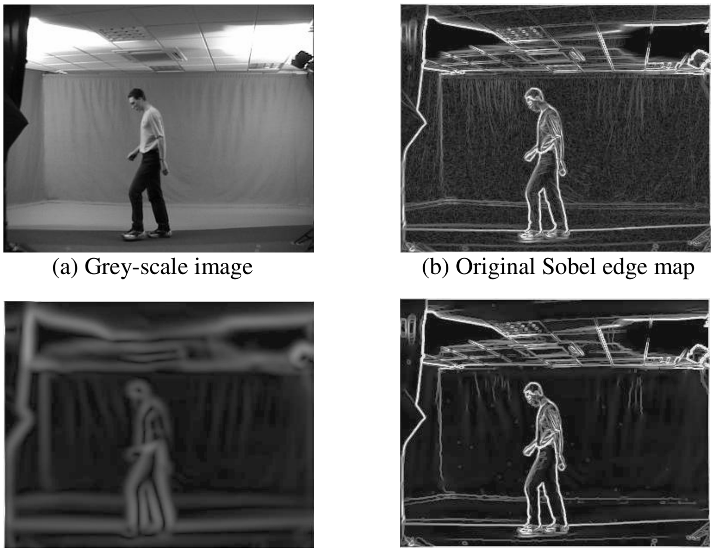
Perona and Malik [4], proposed anisotropic diffusion for selective image smoothing that avoids blurring and localization problems of the edges. The anisotropic heat diffusion equation is given below,

$$\frac{dI}{dt} = div\left(g\left(\left|\nabla I(x,y,t)\right|\right)\nabla I(x,y,t)\right) \tag{3}$$

Where, div represents divergence operator, ∇ is gradient operator, $I(x,y,t)$ is the pixel value of grey level image at location (x,y) and time t, $g\left(\left|\nabla I(x,y,t)\right|\right) \in [0,1]$ is the diffusivity function that is allowed to vary according to the magnitude of the local image gradient ∇I. Different functions are used for $g(.)$ depending on the chosen aim. In our application, an exponential type is used (see Eq. (4)), which prefers high-contrast edges to low-contrast ones.

$$g(\nabla I) = e^{-(|\nabla I|/K)^2} \tag{4}$$

K determines the rate of decay of the exponential function, and thus the rate of smoothing. Note that, if $g\left(\left|\nabla I(x,y,t)\right|\right)$ is constant at all image locations, this leads to isotropic heat diffusion. In Fig. 3, we illustrate the difference between isotropic and anisotropic diffusion operations: Fig. 3(a) is a grey-scale image; and Fig. 3(b) is its Sobel edge map without any diffusion. Fig. 3(c) is the Sobel edge map after isotropic diffusion which causes loss of edge information. On the other hand, Fig. 3(d) is the Sobel edge map of the anisotropic diffused image with diffusivity function given by Eq. (4) and it can easily be observed that high contrast edges are enhanced while removing edges due to noise, and thus important detail is preserved.



(a) Grey-scale image (b) Original Sobel edge map

(c) Sobel edge map after isotropic diff. (d) Sobel edge map after anisotropic diff.

Fig. 3. Difference between isotropic and anisotropic diffusion

4 Isotropic Heat Flow in Temporal Domain

Here, we introduce a novel moving edge detection technique. Once the enhanced Sobel edge maps are obtained in consecutive frames, as shown in Figs. 4(d-f), the isotropic or linear heat equation is applied in the temporal domain to calculate the total amount of heat flow. The discrete form of the isotropic heat equation is given as an iterative process below,

$$E_t^n = E_t^{n-1} + \kappa\Delta E_t^{n-1} = E_t^{n-1} + \kappa(E_{t+1}^{n-1} + E_{t-1}^{n-1} - 2E_t^{n-1}) \tag{5}$$

Where, E_t^{n-1}, E_{t-1}^{n-1} and E_{t+1}^{n-1} are Sobel edge mapped images, after anisotropic diffusion in space, respectively for the reference frame, previous frame and next frame at iteration n. The total amount of heat flow is calculated as follows. Assume that the initial scale is 0 (zero) and final scale is n, then Eq. (5) can be described as,

$$E_t^n = E_t^0 + \kappa\sum_{i=0}^{n-1}\Delta E_t^i \tag{6}$$

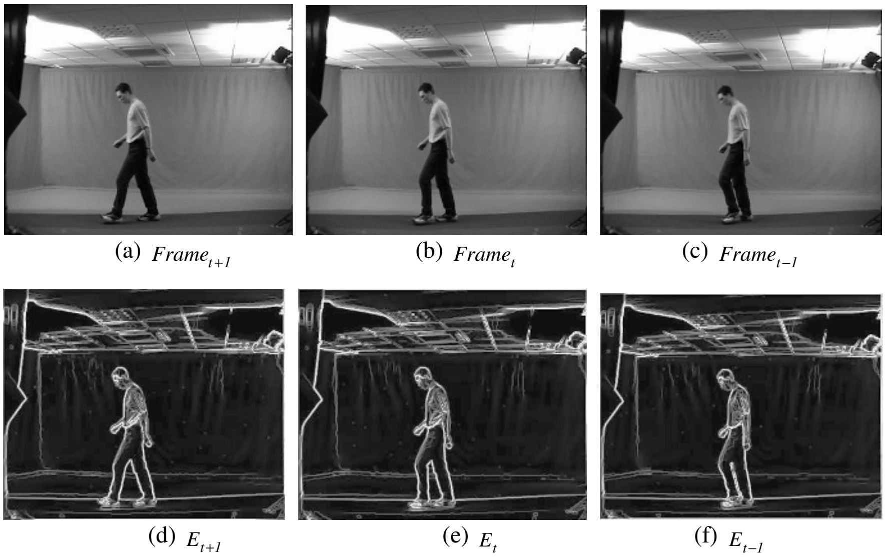
Then, the total amount of heat flow from the initial state to final state is

$$\left|E_t^n - E_t^0\right| = \kappa\sum_{i=0}^{n-1}\left|\Delta E_t^i\right| \tag{7}$$

However this gives us total *heat in* (+) and *heat out* (-) together during diffusion, that is shown in Fig. 4 (g). We are interested in total *heat flow out* (HFO) from the reference frame E_t , which gives us the moving edge map. This is obtained as,

$$HFO = \kappa \sum_{i=0}^{n-1} \left| \Delta E_t^i \right|, \quad \forall \Delta E_t^i < 0 \tag{8}$$

The moving edge map is shown in Fig. 4 (h). Only the moving edges of the human subject and some slight shadow remain, while largely removing the edges introduced by the static background.


(a) $Frame_{t+1}$ (b) $Frame_t$ (c) $Frame_{t-1}$

(d) E_{t+1} (e) E_t (f) E_{t-1}

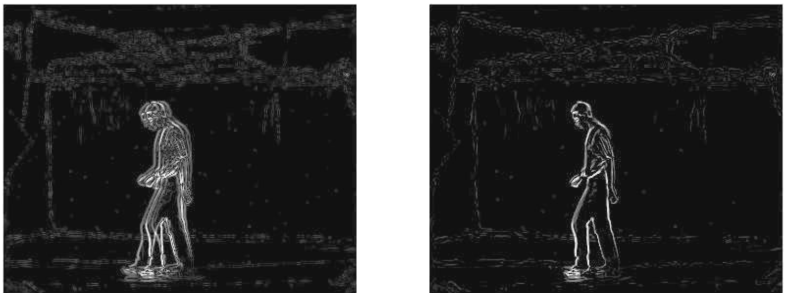

(g) Total heat flow (h) Heat flow out (-)

Fig. 4. Moving edge map extraction

The overall process is completed by non-maxima suppression (thinning) and hysteresis thresholding to observe binary moving edges. The non-maxima suppressed and hysteresis thresholded images are shown respectively in Figs. 5 (a) and (b).

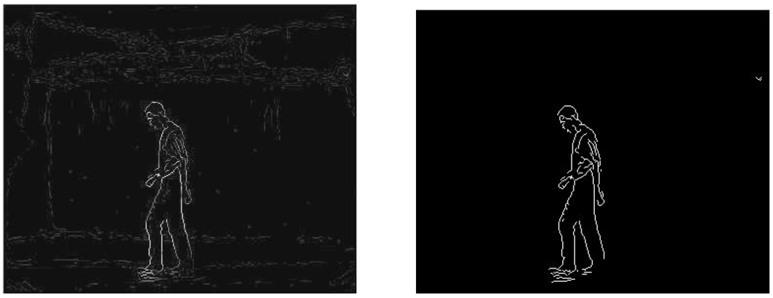

(a) Non-maxima suppressed image (b) Hysteresis thresholded image

Fig. 5. Binary moving edge observation

5 Evaluation and Experimental Results

Performance evaluation is employed by comparing moving edge detection with 2D Sobel edge detection. However, anisotropic heat diffusion in the spatial domain is omitted in our algorithm to balance the 2D Sobel and moving edge detection algorithms. Evaluation is done on a white circle moving on a black background with varying normal distributed noise $N(\mu,\sigma^2)$. The Hough Transform (HT) is applied to the binary edge images to extract circle centre parameters. A root mean square error (RMSE) is then employed to quantify the performance of each algorithm.

$$RMSE = \sqrt{\left((e_x - c_x)^2 + (e_y - c_y)^2\right)/2} \tag{9}$$

Where, (e_x, e_y) are extracted parameters and (c_x, c_y) are actual circle parameters. The quantity of noise is considered in terms of standard deviation σ with zero mean. The threshold for the 2D Sobel, to obtain the binary image, is determined by a root mean square (RMS) estimate of the noise. In this process, the gradient magnitude squared image is thresholded by its scaled mean value that is proportional to signal to noise ratio (SNR). The threshold for 2D Sobel in each noise trial is calculated as,

$$T = s \times \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left|\nabla F_t\right|^2_{i,j} \tag{10}$$

Where, F_t is the reference frame of size $M \times N$, T is the threshold and s is a positive constant, which is 4 in this evaluation. On the other hand, the moving edge detection algorithm has two thresholds (hysteresis thresholding), and their values are based on mean of heat flow out (HFO) image from the reference frame,

$$T_h = c \times \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N} HFO_{i,j}\,, \qquad T_L = T_h / 4 \tag{11}$$

Where, c is a positive constant with value 7 and the ratio between high, T_h, and low, T_L, thresholds is 4. Fig. 6 shows performance of moving edges and 2D Sobel algorithms. It is observed that, until $\sigma \cong 110$ the moving edge detection technique has better performance than 2D Sobel, which appears due to the averaging inherent in the new operator and actually after high noise, input images have very poor quality. Fig. 7 shows some of the results for moving edges (second row) and 2D Sobel (third row). To visual inspection, the input images in Figs. 7(c) and (d), are very noisy indeed.

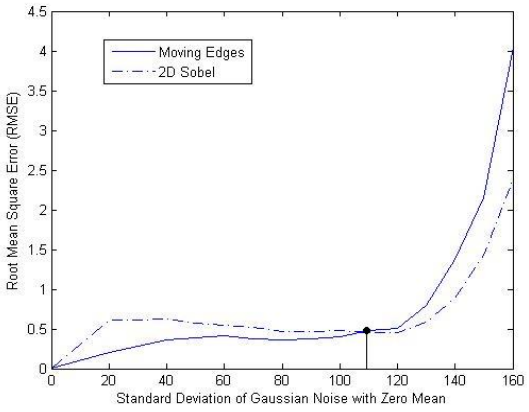
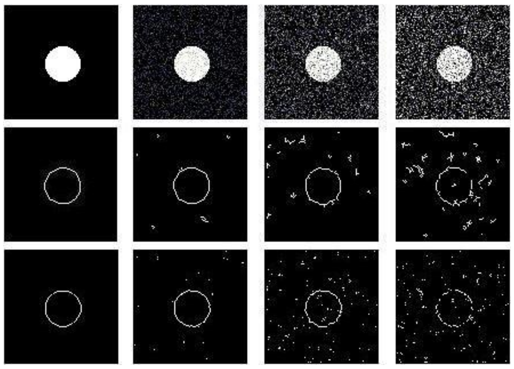


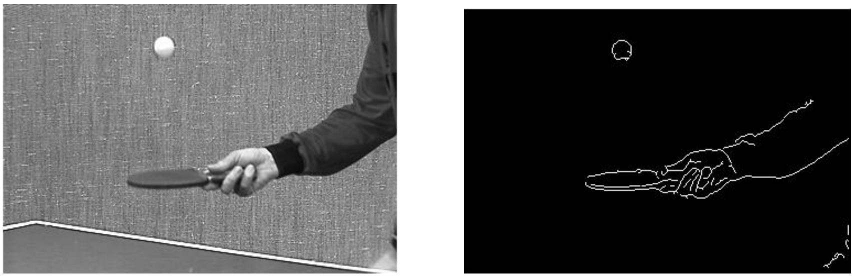
Fig. 6. Performances of moving edges and 2D Sobel with respect to normal distributed noise trials



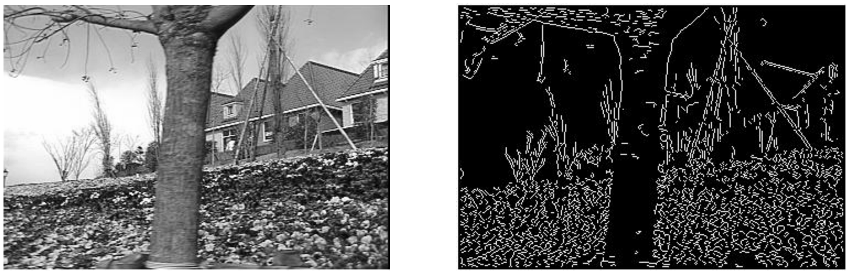
(a) $\sigma = 0$ (b) $\sigma = 40$ (c) $\sigma = 80$ (d) $\sigma = 120$

Fig. 7. Results for moving edges (second row) and 2D Sobel (third row) with respect to increasing Gaussian noise

Simulation results also show that our algorithm can detect moving edges in image sequences, as shown in Fig. 8. Fig. 8(a) shows the reference frame from table tennis (indoor) sequence and moving edges were detected. We should be careful that upper side of the arm and table are stable and this is why they were not detected. Fig. 8(b) is a reference frame from the flower garden (outdoor) sequence, where a camera is in motion. It is seen that moving edges were detected.



(a) Table tennis (indoor) image



(b) Flower garden (outdoor) image, where the camera is in motion

Fig. 8. Some of the simulation results for new operator on indoor and outdoor images

6 Conclusions

We have presented a novel low level moving-feature extraction technique based on heat flow analogy. Firstly, high quality Sobel edge maps are obtained based on anisotropic heat diffusion, in space. The diffusivity function is the key point in this stage; we have chosen an exponential function which enhances high contrast edges and remove edges due to noise. In the next stage, the isotropic heat diffusion is applied in the temporal domain to determine moving edge map in the reference frame. To do this, the total amount of heat flow is calculated and then separated into the heat in (+) and heat out (-) parts, where the heat out (-) is the moving edge map. Finally, non-maxima suppression and hysteresis thresholding is applied to obtain binary moving edges. Evaluation indicates that this technique is better than 2D Sobel until high noise, without anisotropic heat diffusion in space. This result appears due to the averaging inherent in the new operator. Results also show that this technique can detect moving edges in image sequences.

References

1. A. Witkin. Scale-Space Filtering, in Int. Joint Conf. Artificial Intelligence, pp. 1019-1021, Karlsruhe, West Germany, 1983.
2. J. Koenderink. The Structure of Images, in Biological Cybernetics, volume 50, pages 363-370, 1984.

3. R. Hummel. Representations based on zero-crossings in scale-space, in Proc. IEEE Computer vision and Pattern Recognition Conf, pages 204-209, June 1986.
4. P. Perona and J. Malik. Scale-Space and Edge Detection using Anisotropic Diffusion, IEEE Trans. Pattern Analysis and Machine Intelligence, volume 22(8), pages 629-639, 1990.
5. B. B. Kimia and K. Siddiqi. Geometric Heat Equation and Nonlinear Diffusion of Shapes and Images, in Proc. Computer Vision and Pattern Recognition, 1994.
6. F. Zhang and E. R. Hancock. Image Scale-Space from Heat Kernel, Iberoamerican Congress on Pattern Recognition CIARP, pages 181-192, 2005.
7. H. Krim, Y.Bao. Nonlinear Diffusion: A Probabilistic View, Int. Conference on Image Processing, Volume 2, pages 21-25, 1999.
8. O. Scherzer and J. Weickert. Relations between Regularization and Diffusion Filtering, Journal of Mathematical Imaging and Vision, volume 12, pages 43-63, 2000.
9. C.J. Sze and H.Y.M. Liao. A New Image Flux Conduction Model and Its Application to Selective Image Smoothing, IEEE Transaction on Image Processing, Volume 10, No. 2, February 2001.
10. S.T. Acton, A.C. Bovik and M.M. Crawford. Anisotropic diffusion pyramids for image segmentation, Proc. First IEEE Int. Conf. Image Processing, Austin, November 1994.
11. S. Manay and A. Yezzi. Anti-Geometric Diffusion for Adaptive Thresholding and Fast Segmentation. IEEE Transaction on Image Processing ,Volume 12, No.11, November 2003.
12. X. Ji and J. Feng. A New Approach to Thinning Based on Time-Reversed Heat Conduction Model, IEEE Int. Conf. Image Processing, Volume 1, pages 653-656, October 2004.
13. M. Kass, A. Witkin and D. Terzopoulos. Snakes: Active Contour models, International Journal of Computer Vision, pages 321-331, 1987.
14. C. Xu and J.L. Prince. Snakes, Shapes and Gradient Vector Flow, IEEE Transaction on Image Processing, Volume 7, No. 3, pages 359-369, March 1998.
15. V. Caselles, F. Catte, T. Coll and F. Dibos. A Geometric Model for Active Contours. Numerische Mathematic, Volume 66, pages 1-31, 1993.
16. R. Malladi, J.A. Sethian and B.C. Vemuri. Shape Modeling with Front Propagation: A Level Set Approach. IEEE Transaction on Pattern Analysis and Machine Intelligence, Volume 17, No. 2, pages 158-175, 1995.
17. B. Horn and B. Schunck. Determining Optical Flow, Artificial Intelligence, Volume 17, pages 185-204, 1981.
18. J. Weickert and C. Schnorr. Variational Optic Flow Computation with a Spatio-Temporal Smoothness Constraint,Journal of Mathematical Imaging and Vision, Volume 14, pages 245-255, 2001.
19. L. Alvarez, J. Weickert and J. Sanchez. A Scale-Space Approach to Nonlocal Optical Flow Calculations, Scale-Space theories in Computer Vision, 1999.
20. S.K. Makrogiannis and N.G. Bourbakis. Motion Analysis with Application to Assistive Vision Technology, 16th IEEE International Conference on Tools with Artificial Intelligence, pages 344-352, 2004.
21. J.H. Lienhard IV and J.H. Lienhard V. A Heat Transfer Textbook, 3rd Edition, Phlogiston Press, Cambridge Massachusetts, 2005.

Shape Tracking and Registration for 4D Visualization of MRI and Structure

Irene Cheng, Sharmin Nilufar, Anup Basu, and Randy Goebel

Dept. of Computing Science, University of Alberta, Alberta, Canada
lin@cs.ualberta.ca

Abstract. We describe our preliminary research on integrating MRI video with a 3D surface scan of a face. Our approach first extracts contours of a video by using snakes [4, 5]; then the outline structure of the video is matched with a close matching contour on the 3D face structure. The matching and alignment of the two representations uses curvature representations along with some simple heuristics about the relative locations of the facial features, such as nose and chin.

Even though techniques like video fluoroscopy [8] can create high quality images, it subjects patients to high volumes of radiation, and cannot be used to monitor patients over short time intervals. Our alternative combines MRI video with 3D facial structure to improve visualization for medical professionals. The MRI video was created in our related research [7] by registering multiple MRI sequences of swallowing.

1 Introduction

MRI capture does not subject patients to harmful radiation and is thus preferable over CT and video fluoroscopy. However, a limitation in MRI is the relatively low resolution and the inability to view this information in conjunction with 3D structure in order to visualize internal images with a reference to the external structure.

Here we attempt to integrate MRI video with 3D facial structure to obtain a 4D visualization environment. The steps needed to achieve this include MRI video interpolation described in our related research, face contour tracking in MRI video, feature detection in the face contour, and finally, alignment of the video plane with the 3D face structure. The 3D facial texture (Figure 1a), structure (Figure 1b), and after texture mapping (Figure 1c) was captured using a 3D scanner. Figure 1d shows a frame from an MRI video sequence. Our challenge is to be able to view the MRI video registered and properly aligned on a planar surface inside the 3D face structure.

The remainder of this report is organized as follows: Section 2 describes a strategy for MRI video contour tracking using active contours and feature detection using curvature; in Section 3 we discuss how the MRI contour can be matched with a 3D structure contour for integrated MRI video and 3D structure visualization; finally, Section 4 provides concluding remarks and an outline of future.

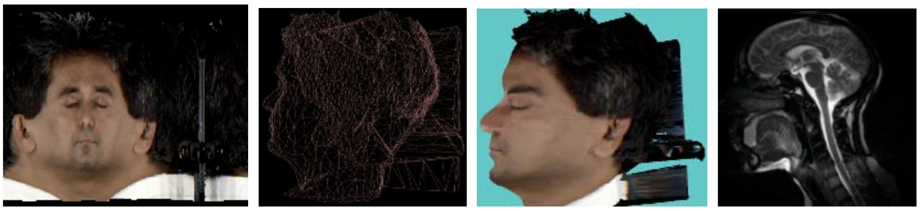


Fig. 1. (a) Scanned texture; (b) & (c) Mesh and Texture-mapped structure, respectively, with support on back of head; and (d) A frame of the MRI video

2 Video Contour Tracking

We used an active contour model [4, 5] to detect the boundary of MRI face slices, followed by automatic detection of some face features: *e.g.*, nose tip and chin using curvature on the MRI contour. We decided to use *snake* for the video contour tracking method in the current implementation because of a number of attractive properties which make snake more efficient for contour detection than other boundary extraction methods. For example, many contour extraction techniques, *e.g.* edge detection and linking, region growing, and relaxation labeling, use local edge information and they must find suitable constraints for continuity when edges are broken. Unless special constraints are predefined, incorrect boundaries can be extracted. Furthermore, these algorithms are generally constrained by the resolution of the images and often generate inaccurate results. Snake or the active contour model overcomes this problem because the contour connectivity is a part of the model and the global content of the image is taken into consideration when detecting boundaries. Snake is rotation invariant, and can also effectively track contours in a sequences of images when the sampling interval is small.

The basic snake model proposed by Kass *et al.* [4] has two major limitations. First the initial contour must be sufficiently close to the true boundary; otherwise it will converge to a wrong result. Second, snakes have difficulties progressing into concave boundary regions. A new type of snake, called the gradient vector flow (GVF) snake proposed by Xu *et al.* [9], effectively addresses both difficulties. The advantage of using GVF snake is that it has large capture range and is able to move into concave boundaries.

2.1 Gradient Vector Flow (GVF) Snake:

The two major limitations of the basic snake model were related to the contour initialization and poor convergence to the concave boundary. In the enhanced model [9, 10], a new static external force field was introduced, which replaced the standard external force field in the traditional snake.

The new static external force is called *gradient vector flow force* and is defined by $F_{ext}^{(g)} = V(x,y)$, which does not change with time, nor depend on the position of the snake itself. Replacing the original external potential force with *V* we obtain the following equation:

$$x_t(s,t) = \alpha x''(s) - \beta x'''(s) + V$$

The parametric curve solving the above dynamic equation is called a GVF snake. Standard numerical methods can be used to solve this equation.

The *gradient vector flow field* is defined as the vector field $V(x, y) = (u(x, y), v(x, y))$ that minimizes the following energy functional:

$$E = \iint \mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + |\nabla f|^2 |v - \nabla f|^2 \, dxdy$$

Where *f(x,y)* is the edge map of the image. The parameter μ is a regularization parameter that adjusts the tradeoff between the first and second terms of the integrand, and is set according to the level of noise present in the image. In addition, where the value of the edge gradient is small, energy is dominated by the sum of the partial derivatives of the gradient field. When the gradient is large, the second term dominates.

2.2 Curvature Calculation

The degree of curvature can be determined by computing the rate of change of the surface tangents. A line should have zero curvature, a curve with very shallow concavity should have a small number to represent its curvature, and a sharp corner should have a relatively larger number to represent its curvature [7, 11]. In our implementation we wanted to find the tip of a nose and chin by calculating the local maximum curvature around the nose and chin location using the following curvature formula.

Let γ(t) be the position vector in the area under examination. Curvature at γ(t) can be characterized by the unit tangent vector T(t), unit normal vector N(t), or the radius of the osculating circle. The position vector $\gamma(t) =< x(t), y(t) >$ generates a path on the plane over time t. To capture the velocity vector, we take the derivative of the position vector with respect to time t.

$v = \dfrac{d\gamma}{dt} = \gamma'(t) =< x'(t), y'(t) >$. The unit tangent vector is given by:

$$T = \frac{\gamma'(t)}{\|\gamma'(t)\|}$$

If a curve is parameterized in terms of arc length *s*, then the curvature can be computed as:

$$\kappa = \left\| \frac{dT}{dt} . \frac{ds}{dt} \right\| = \frac{1}{\|v\|} . \left\| \frac{dT}{dt} \right\|$$

2.3 Outline of the Algorithm

The following steps were performed to extract the head contour in a MRI slice:

1) Read the MRI sequence in "DICOM" format (using dicomread function in Matlab).
2) Enhanced the images using histogram equalization and contrast stretching.

3) Output the data in JPEG format (using imwrite function in Matlab).
4) Compute the edge map of the MRI slices and then normalized the intensities of the edge map to the range [0, 1].
5) Used the normalized edge intensities as the input to the GVF solver to produce the GVF field.
6) Initialized the 2-D snake manually on the user interface.
7) Repeated the following steps for a number of iterations until a statistical equilibrium is reached:
    (a) Deformed snake in the given external force field.
    (b) Interpolated the snake adaptively.
8) Applied the contour detected in the first slice as the initial contour or seed contour in the next slide and so on, to find the face boundary for the remaining MRI face slices.
9) Finally, detected the nose tip and chin by using the following steps:
    (a) Found the nose and chin location by windowing;
    (b) Fitted polynomial curves for the nose and chin regions;
    (c) Calculated the maximum curvature in the nose and chin areas.

2.4 Experimental Results

The original image sequence contains 32 MRI slices in dicom format, which had very low contrast. A significant amount of preprocessing was required to enhance the contrast before applying the contour detection method. Figure 2 shows an example of (a) the face image after applying histogram equalization, (b) the face image after contrast stretching, (c) the final convergence of the GVF snake, and (d) the detected face contour.

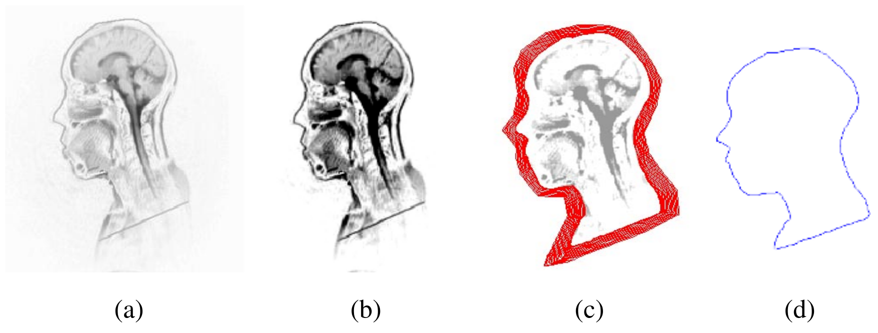


Fig. 2. (a) image after histogram equalization; (b) after further contrast enhancement; (c) Convergence of GVF snake; and (d) final result

Snake generation was implemented in Matlab. The basic Matlab program of GVF snake was downloaded from the author's website http://iacl.ece.jhu.edu/projects/gvf/. Several parameters of the GVF active contour model such as elasticity, rigidity, viscosity and external force weight were modified in order to achieve better results for our application. The convergence time of the snake depends on the number of iterations. 125 iterations were required in our experiment, taking about 80 seconds.

Contour detection in successive MRI slices

In order to track the face contours in a sequence of 32 MRI slices, the snake was initialized to track the contour on the first MRI slice. The contour detected in the previous slice was then used as the seed to detect the contour of the subsequent slice, and so on. The result showed that the GVF snake was able to conform to the correct object boundaries in the MRI sequence using the contour of the previous slice. Figure 3 (left) shows how the snake converged when we used the contour detected in the previous slice as the seed. Since there is only a slight difference in the contours between two consecutive slices, it is difficult to view the convergence process and to differentiate the boundary of the two slices. Figure 3 (right) shows the final contour for the second MRI slice.

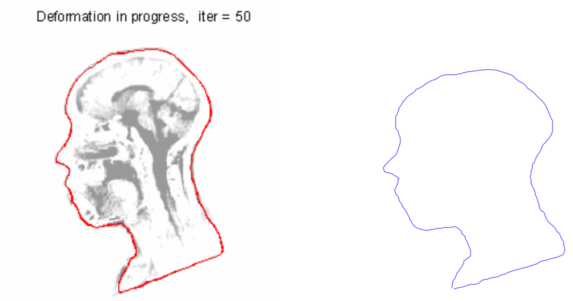


Fig. 3. (Left) shows the convergence of GVF snake in two successive slice; (Right) the detected contour in the second MRI slice.

Detection of nose tip and chin

The implementation of the curvature calculation algorithm was applied to the detected contour in order to find the local extrema, *e.g.* nose tip and chin tip. Before calculating the curvature we tried to fit a polynomial to the detected nose and chin boundary. The best results were obtained by using 11 points for the chin region and fitting a polynomial of degree 3, and selecting 4 points in the nose area and fitting a polynomial of degree 3. Figures 4 (left) and 4 (right) show the original contour points (marked by stars) and fitted curves (represented by dotted line) for nose and chin, respectively.

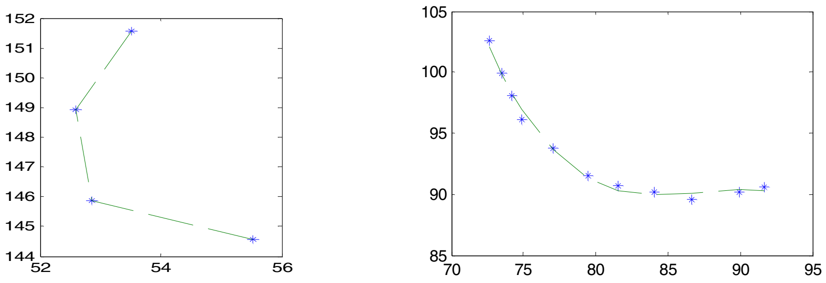


Fig. 4. Fitted curves for (left) nose and (right) chin areas

Figure 5 (a) shows the result obtained from the curvature calculation to find the nose tip and chin automatically. To show that the method is invariant to rotations, we detected the nose and chin after rotating the face at different angles as shown in Figures 5 (b) to 5 (f).

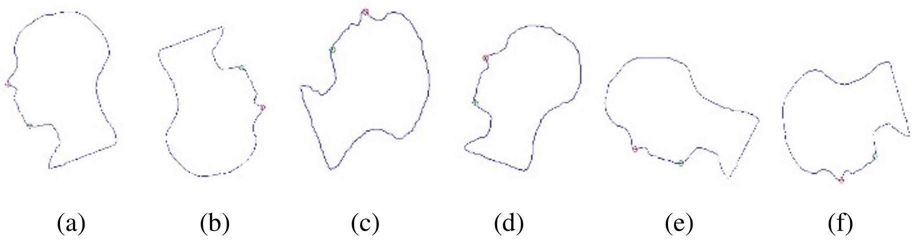


Fig. 5. (a) shows automatic detection of nose tip and chin by calculating maximum curvature; (b)-(f) show the detection results at different orientations of the face

Performance comparison with traditional snakes

Figure 6 (left) shows the convergence of a traditional snake. Figures 6 (middle and right) show the convergence results after 500 and 1000 iterations (compared to only 125 iteration using GVF). We can also notice the poor convergence result of the traditional snake on boundary concavities, *e.g.* in the throat area.

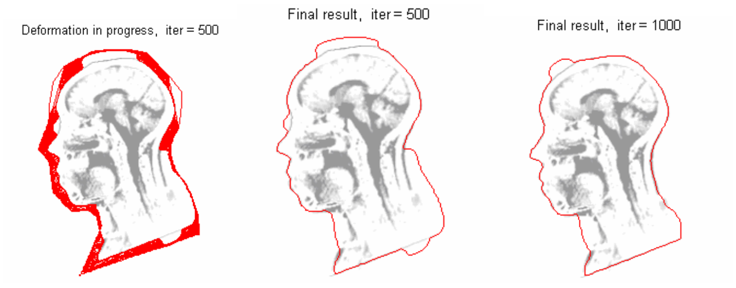

Deformation in progress, iter = 500
Final result, iter = 500
Final result, iter = 1000

Fig. 6. (Left) Convergence of traditional snake. (Middle) and (right) show the detected contours with traditional snake after 500 and 1000 iterations.

3 Matching Video Contour with 3D Contour for Integrated Visualization

So far, we have implemented a preliminary approach for matching MRI video contour to the 3D contour of a face. To the extracted video contour we fit local 4th degree polynomials and calculate curvatures along the contour. When matching curvature over time, it is important to choose features that are invariant to time. Since our focus is on the human face, the best or most prominent feature selection is along the contours from the forehead to the noise tip, and to the chin (Figure 7). We use the Cyberware laser scanner to capture the head geometry, but hair and ear are difficult or impossible to capture using laser scanning, and should not be used for matching.

From the face contours, using simple heuristics such as sharpness of the nose and chin and their relative locations, we extract reliable features (Figure 5) that are invariant to the orientation changes of the contour.

Alignment of the MRI video with the 3D structure is achieved by detecting and aligning corresponding features, like nose and chin tips, on the MRI video contour and the contour generated by a vertical slice of scanned 3D surface points.

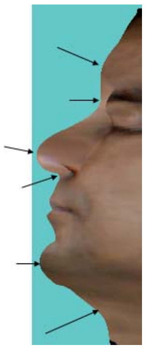


Fig. 7. Example of prominent contours on a human face

The traditional approach to analyzing medical images relies on two-dimensional images, *e.g.* X-ray mammography. In the last ten years, analysis on three-dimensional models in medical applications has received significant attention (Figure 8 (a) and (b)). Although 3D models are more visually appealing, there are two limitations: First, they often do not capture the photorealistic texture with high resolution. Per-vertex color is often used but not every vertex is associated with a real color; most of the algorithms assign colors interpolated from neighboring vertices. In addition, gray-scale is used as the color space. Second, 3D static models are not time-varying data. They do not trace the flow of an activity, *e.g.* swallowing.

In our four-dimensional (4D) approach, we incorporate the time varying data, in the form of video, into the 3D model, so that a sequence of soft-tissue or muscle movements can be analyzed by the physician. The surface data can be shown as a mesh or a transparent skin, allowing the physician to visualize the inside structure, without losing the facial expression of the patient (Figure 8 (c)). The CT scan is better for capturing bone structures and the MRI scan is better for capturing soft-tissue. In the traditional 3D modeling approach, MRI data (Figure 8 (a)) cannot be visualized together with the skin complexion. Although both soft-tissue (brain) and bone structure are shown in Figure 8 (b), half of the skull has to be removed in order to expose the brain. In our 4D approach (Figure 8 (c)), the skin complexion and soft-tissue can be visualized at the same time, displaying the movement of the brain tissue using a video sequence. A layered 4D model can also be used with skin as the outer layer, bone structure as the second layer, and soft-tissue as the inner layer. However, there is a data format problem to be considered: 3D data format for medical applications follows the Digital Imaging and Communications in Medicine (DICOM) standard for distributing and viewing. The basic structure of DICOM is composed of blocks (voxels) filling the interior of a volume. The smaller the blocks, the better is the representation of the silhouettes of the 3D structure and the higher the precision. In order to maintain precision, the amount of DICOM data generated from CT and MRI scans are therefore very large.

In order to adopt the 4D visualization model, or layered 4D model, volume data has to be converted to surface data. A set of 3D points can be extracted by taking the centers of the blocks. If we extract the centers of the blocks lying only on the

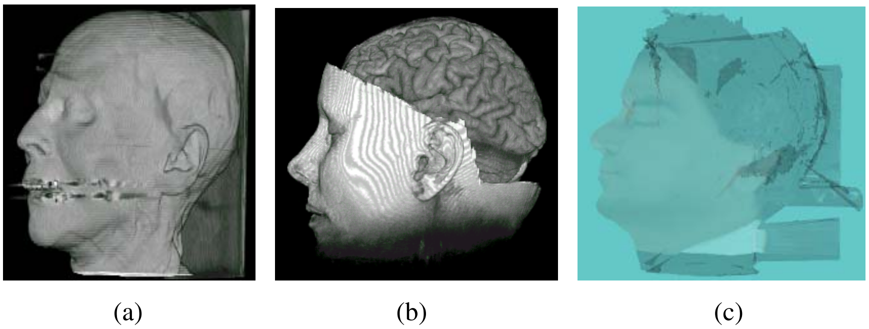


Fig. 8. (a) MRI of a human head with skull removed, and (b) CT scan of a head, both from UNC data set, displayed using Sun Microsystems Java 3DVolume demo program; and (c) Transparent skin presenting facial expression of the patient as well as the required images inside the surface structure

boundary, the surface of the point set can be generated. The surface points can then be triangulated to form a surface mesh. There are far fewer modeling algorithms on volume data than compared with those developed for mesh data. By converting a volume space into a mesh space, more processing techniques can be considered to achieve desired results. Another advantage of using the mesh space is to generate a hierarchy of 3D models (Figure 9) [1, 2, 3]. Depending on the precision and application requirements, the appropriate level-of-detail (LOD) can be selected for analysis.

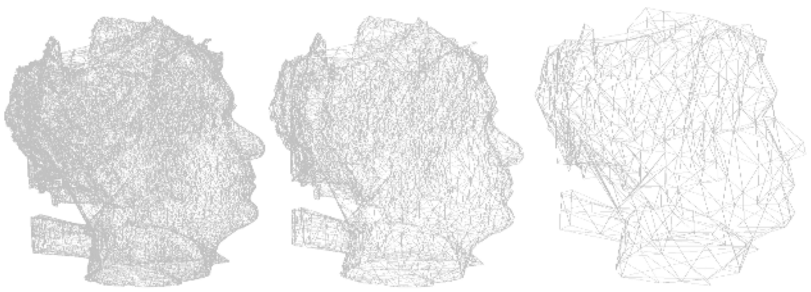


Fig. 9. Different level-of-details of the head structure. From left to right: 50,000 faces, 10,000 faces, and 1,000 faces.

A sequence of slices can be extracted from a mesh structure by intersecting the mesh with a set of parallel planes horizontally, vertically, diagonally, or in other orientations. We divide the head structure into slices consistent with the viewpoint of the video. Each video sequence is then mapped onto the corresponding slice (Figure 10).

Figure 11 shows screen captures at various time points of the integrated 3D surface along with the MRI video plane. Figure 11 (a) shows the scanned 3D structure with support on back of the head; Figure 11 (b) with support frame part of mesh removed; and Figure 11 (c) shows a close up.

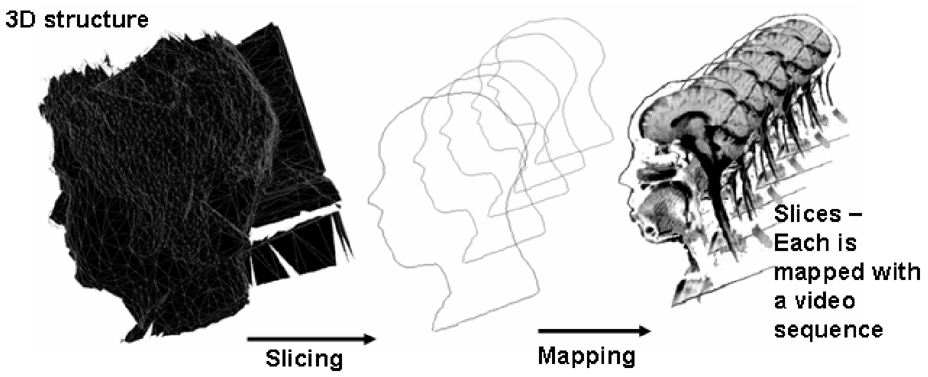


Fig. 10. A block diagram showing the different stages of video sequence mapping for 4D visualization

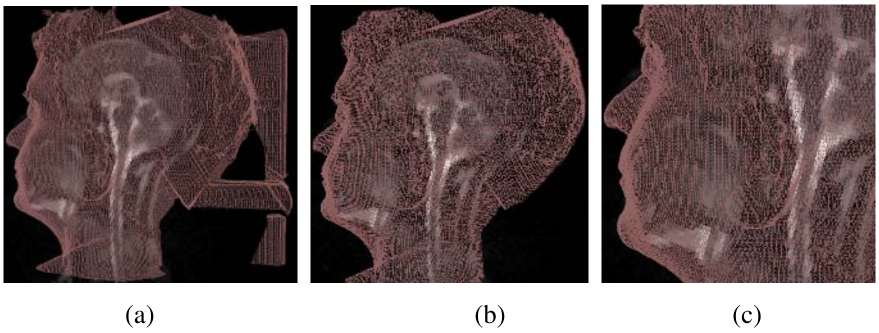


Fig. 11. Snapshots of integrated MRI video inside 3D face model

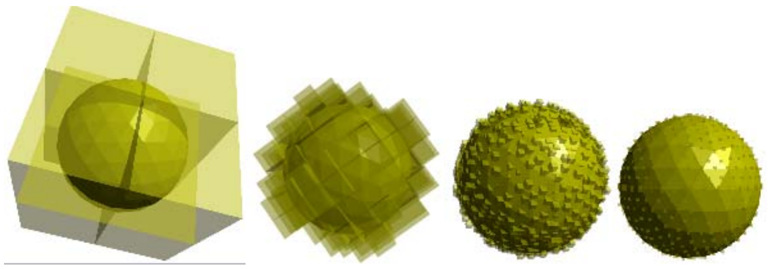


Fig. 12. A spherical tumor is partitioned into half in each of the x-, y- and z- directions (in 1 iteration). From left to right: the results after 1, 3, 5 and 6 iterations are shown.

If necessary, mesh data can also be converted back to the volume space, by filling the interior of the mesh with blocks. For example, Figure 12 illustrates how the mesh of a spherical tumor can be converted to a volume space. Our algorithm works as follows:

1. The mesh is enclosed in a minimum cube (block).
2. The block is divided into eight sub-blocks using three orthogonal planes (in 1 iteration).
3. Any empty block outside the mesh is thrown away.
4. Steps (2) and (3) are repeated until the desired number of iterations is reached.

The flexibility of interchanging between volume and surface space enables medical data to be modeled and analyzed taking advantage of the techniques developed for different data spaces.

4 Conclusion and Future Work

Active contour models play a vital role in determining any abnormalities found in biomedical imaging modalities, because these models are able to detect the complex shapes of anatomical structures successfully. In this work we used the gradient vector flow based active contour generation algorithm to detect the contour from MRI image slice. Following this step, we outlined procedures for feature detection on a plane of a 3D structure and integrated 4D visualization of structure and MRI video.

The MRI video planes are assumed to be vertical and saggital planes with respect to the 3D surface scan in Figure 1. This assumption may not produce good results when the actual plane could be slightly tilted from the vertical, because of the difficulty in having accurate head positioning and alignment during MRI or 3D scans. In future work we will look into the more difficult registration problem that addresses the variability of the orientation of the video plane.

References

[1] I. Cheng and P. Boulanger, "Feature extraction on 3D texMesh using scale-space analysis and perceptual evaluation," IEEE Trans. on Circuit and Systems for Video Technology Special Issue, Vol. 15, No. 10, 1234-1244, October 2005.

[2] M. Garland and P. Heckbert, "Simplification using Quadric Error Metrics," In Proc. SIGGRAPH 1997, pp.209-216.

[3] H. Hoppe, "Progressive Meshes," In Proc. SIGGRAPH 1996, pp. 99-108.

[4] M. Kass, A. Witkin, and D. Terzopoulos, Snakes - Active Contour Models, International Journal of Computer Vision, 1(4): 321-331, 1987.

[5] F. Leymarie and M. D. Levine, "Tracking deformable object in the plane using an active contour model," IEEE Trans. Pattern Anal. Machine Intelligence., vol. 15, no. 6, pp. 617–634, June 1993.

[6] D. Salomon, "Curves and Surfaces for Computer Graphics," Springer, 2006.

[7] Has to be anonymous for now, to appear in a conference proceedings.

[8] R.E.R. Wright, C.S. Boyd and A. Workman, Radiation doses to patients during pharyngeal videofluoroscopy, 13: 113-115, Dysphagia 1998.

[9] C. Xu and J.L. Prince, Gradient Vector Flow: A New External Force for Snakes, Proc. IEEE Conf. on Comp. Vis. Pattern Recognition (CVPR), Los Alamitos: Comp. Soc. Press, pp. 66-71, June 1997.

[10] C. Xu and J.L. Prince, Snakes, Shapes, and Gradient Vector Flow, IEEE Transactions on Image Processing, 7(3), pp. 359-369, March 1998.

[11] http://online.redwoods.cc.ca.us/instruct/darnold/MultCalc/Curvature/context-curvature-p.pdf

History Trees as Descriptors of Macromolecular Structures

Deniz Sarioz[1], T. Yung Kong[2], and Gabor T. Herman[1]

[1] Ph.D. Program in Computer Science, The Graduate School and University Center
City University of New York, New York, NY 10016, USA
sarioz@acm.org, gabortherman@yahoo.com
[2] Department of Computer Science, Queens College,
City University of New York, Flushing, NY 11367, USA
ykong@cs.qc.edu

Abstract. High-level structural information about macromolecules is now being organized into databases. One of the common ways of storing information in such databases is in the form of three-dimensional (3D) electron microscopic (EM) maps, which are 3D arrays of real numbers obtained by a reconstruction algorithm from EM projection data. We propose and demonstrate a method of automatically constructing, from any 3D EM map, a topological descriptor (which we call a history tree) that is amenable to automatic comparison.

Keywords: Discrete shape representation, digital topology, macromolecular structures, volume images.

1 Introduction

High-level structural information about macromolecules is now being organized into databases such as the Quaternary Protein Structure (QPS) and the Electron Microscopy Data Base (EMDB), both at the European Bioinformatics Institute (EBI). Initiatives in the EM field are also starting in the US, nucleated around the Research Collaboratory for Structural Bioinformatics (RCSB) that is responsible for the database called the Protein Data Bank (PDB).

These databases include reconstructions from EM data, i.e., 3D arrays of real numbers that are voxelizations of macromolecular structures. Suppose that a biology researcher has obtained from EM projections a new reconstruction of the structure of a macromolecule, and would like to see if a database contains a similar object. The very large size of these 3D arrays, the arbitrary position and orientation of the molecule in the array and the possibility of non-linear stretching of the range make standard methods of comparison infeasible. Hence, there is a need for exploring and analyzing topological and geometrical features of the contents of such large aggregates in a systematic, quantitative and automatic manner to mine the information contained in these databases. In the following we propose a method of automatically producing topological descriptors of 3D EM arrays, and demonstrate it on arrays that we acquired from EBI. We believe that comparison of these descriptors using appropriately defined similarity measures will be useful in the identification and classification of macromolecules.

2 Mathematical Preliminaries

2.1 Foreground Components in 3D Images; *f*-Ancestors

An EM reconstruction is typically represented as a 3D image — i.e., a real-valued mapping f defined on the voxel set X of a box-shaped region. For every real number t, we define the *foreground voxel set* of f at t as: $\mathcal{F}_f(t) = \{x \in X \mid f(x) \geq t\}$. Note that $\mathcal{F}_f$ shrinks monotonically: if $t_1 < t_2$, then $\mathcal{F}_f(t_1) \supseteq \mathcal{F}_f(t_2)$. We partition each foreground voxel set $\mathcal{F}_f(t)$ into connected components based on 6-connectivity [1]. [Two voxels are 6-*adjacent* if they share a face. A 6-*path* in U is a sequence of voxels in U in which every consecutive pair of voxels are 6-adjacent. A set U of voxels is 6-*connected* if for every pair of voxels in U there is a 6-path in U that begins at one voxel of the pair and ends at the other. A 6-*component* of U is a maximal non-empty 6-connected subset of U.]

Let $C_f(t)$ denote the collection of 6-components of $\mathcal{F}_f(t)$ and let $\mathcal{C}_f = \bigcup_{t \in \mathbb{R}} C_f(t)$. If $D_1, D_2 \in \mathcal{C}_f$, then we say D_1 is an *f-ancestor* of D_2 if there exist $t_1 \leq t_2$ such that $D_1 \in C_f(t_1)$, $D_2 \in C_f(t_2)$ and $D_1 \supseteq D_2$.

2.2 Foreground History Trees

We define the *ancestor* and *descendant* relations on the vertices of a rooted tree recursively, as follows (cf. [2, p. 93]): If v_1 and v_2 are vertices of a rooted tree, then v_1 is an ancestor of v_2 (and v_2 is a descendant of v_1) if $v_1 = v_2$ or v_1 is an ancestor of the parent of v_2.

A *foreground history tree* (FHT) for an image f is a rooted tree T in which each vertex v is associated with a real number $L_T(v)$, called its *level*, and a set $D_T(v) \in C_f(L_T(v))$, and in which the following conditions are satisfied:

1. If $L_T(v_1) = L_T(v_2)$ and $D_T(v_1) = D_T(v_2)$, then $v_1 = v_2$.
2. A vertex v_1 is an ancestor in T of a vertex v_2 if, and only if, $L_T(v_1) \leq L_T(v_2)$ and $D_T(v_1)$ is an f-ancestor of $D_T(v_2)$.

We will usually omit the subscript T from L_T and D_T unless it is needed to distinguish the L and D functions of different FHTs.

Note that if a vertex v_1 is the parent of a vertex v_2 in an FHT, then we have that $L(v_1) < L(v_2)$. Indeed, v_1 is an ancestor of v_2, and so $L(v_1) \leq L(v_2)$ and $D(v_1)$ is an f-ancestor of $D(v_2)$. Suppose $L(v_1) = L(v_2)$. Then $D(v_1)$ and $D(v_2)$ would both be 6-components of $\mathcal{F}_f(L(v_1)) = \mathcal{F}_f(L(v_2))$, and since $D(v_1) \supseteq D(v_2)$ (because $D(v_1)$ is an f-ancestor of $D(v_2)$) this would imply $D(v_1) = D(v_2)$. But then $v_1 = v_2$ (as $L(v_1) = L(v_2)$) contrary to the fact that v_1 is the parent of v_2.

FHTs are related to contour trees [3]; but contour trees are not necessarily rooted, whereas FHTs are. It is our hypothesis that FHTs can be made to reflect certain essential properties of macromolecules, and so provide a suitable basis for assessing the similarity of macromolecules.

3 Methods and Results

3.1 Obtaining a Foreground History Tree

Let $f : X \to \mathbb{R}$ be a 3D image, and let $\tau_1 > \tau_2 > \ldots > \tau_k$ be a strictly decreasing finite sequence of real numbers such that $f_{\max} \geq \tau_1$ and $f_{\min} \geq \tau_k$, where $f_{\max}$ and $f_{\min}$ are the maximum and the minimum values attained by f on the set of voxels X. [Thus $\emptyset \subsetneq \mathcal{F}_f(\tau_1) \subseteq \mathcal{F}_f(\tau_2) \subseteq \ldots \subseteq \mathcal{F}_f(\tau_k) = X$, and $|C_f(\tau_k)| = 1$.]

We now describe an algorithm that constructs an FHT T for f which satisfies the condition $\{L(v) \mid v \in \text{Vertices}(T)\} = \{\tau_i \mid 1 \leq i \leq k\}$, and which has as many vertices as is possible for such an FHT. For convenience in describing the algorithm, we define τ_0 to be an arbitrary but fixed number that exceeds $f_{\max}$, so that $\mathcal{F}_f(\tau_0) = \emptyset$ and hence $C_f(\tau_0) = \emptyset$.

The algorithm is based on a data structure which represents a collection of pairwise disjoint nonempty sets of voxels. Operations MAKE_SET(x) and UNITE_SETS(x_1, x_2) are used to change the represented collection of sets. These are defined below the next paragraph, whose aim is to provide the reader with an initial understanding of how the algorithm alters the data structure and, at the same time, builds up the FHT.

The algorithm has a main loop whose body is executed k times. At the end of the ith iteration of the loop, the data structure represents the collection of sets $C_f(\tau_i)$ and the FHT has been built (from its leaves towards its root) up to the level τ_i; i.e., all vertices v in the eventual FHT for which $L(v) \geq \tau_i$ have been created, and both $L(v)$ and $D(v) \in C_f(L(v))$ have been determined for these vertices, as well as the parent-child relationships among them. Clearly, at the end of the kth iteration, we have produced the whole FHT.

When the data structure represents a collection $\mathcal{S}$ (of disjoint sets of voxels), and x is any voxel such that $x \notin \bigcup \mathcal{S}$, a call of MAKE_SET($x$) adds the singleton set $\{x\}$ to the represented collection of sets: It changes the data structure from a representation of $\mathcal{S}$ to a representation of $\mathcal{S} \cup \{\{x\}\}$.

The algorithm calls UNITE_SETS either to replace two existing sets of voxels in the represented collection of sets with the union of those two sets, or to insert a new voxel, which does not yet belong to any set in the represented collection, into one of the sets in the collection. The effect of calling UNITE_SETS can be more precisely described as follows. Let x_1 and x_2 be voxels, and let $\mathcal{S}$ be the collection of sets that is represented by the data structure when UNITE_SETS(x_1, x_2) is called. For $i = 1, 2$, let $S_i = \{x_i\}$ if $x_i \notin \bigcup \mathcal{S}$, and let S_i be the member of $\mathcal{S}$ that contains x_i if $x_i \in \bigcup \mathcal{S}$. Then the call UNITE_SETS(x_1, x_2) changes the data structure from a representation of $\mathcal{S}$ to a representation of the collection $(\mathcal{S} \setminus \{S_1, S_2\}) \cup \{S_1 \cup S_2\}$.

The data structure is in fact a forest of rooted trees of nodes, together with a list of the root nodes of all the trees in the forest. Our data structure is an example of a *disjoint-set forest* (DSF) [2, pp. 446–450]. Trees and nodes of our DSF will be called DSFtrees and DSFnodes. The list of root nodes is given by the variable CURRENT_DSF_ROOTS, which is updated by UNITE_SETS and MAKE_SET.

There is a 1-to-1 correspondence between the collection of all DSFtrees of the DSF and the collection of disjoint sets of voxels that is represented by the DSF. Each voxel x has a field x.DSFnode that can either be a DSFnode or be NIL. A voxel x belongs to the set of voxels that corresponds to a DSFtree $\mathcal{T}$ if, and only if, x.DSFnode is a DSFnode in $\mathcal{T}$. Thus x.DSFnode is NIL if, and only if, x does not belong to any set in the collection of sets that is represented by the DSF. If x.DSFnode is NIL, then a call of MAKE_SET(x) will create a new DSFtree that consists of one new DSFnode and will set x.DSFnode to be that new DSFnode. This is the only way in which the algorithm creates DSFnodes.

The role of a DSFnode in our algorithm is analogous to the role of a label in standard connected component labeling algorithms for binary images (see, e.g., [4, pages 347–349]). DSFnodes that belong to the same DSFtree correspond to "equivalent" labels (i.e., labels that represent the same component).

The algorithm uses a function DSF_ROOT(), which is such that if ν is any DSFnode then DSF_ROOT(ν) returns the root of the DSFtree which contains ν. Thus two voxels x and y belong to the same member of the collection of sets that is represented by the DSF if, and only if, x.DSFnode $\neq$ NIL, y.DSFnode $\neq$ NIL and DSF_ROOT(x.DSFnode) = DSF_ROOT(y.DSFnode).

The algorithm starts by scanning the voxels in X and assigning each of them to one of k voxel "bins" $B[1], B[2], \ldots, B[k]$ as follows: A voxel $x \in X$ is placed in the bin $B[j]$, where j is the integer such that $\tau_{j-1} > f(x) \geq \tau_j$. [Thus the set of voxels that are placed in each bin $B[i]$ is just $\mathcal{F}_f(\tau_i) \setminus \mathcal{F}_f(\tau_{i-1})$.] The DSF is initialized to be empty, so x.DSFnode is NIL for every voxel x in X, and CURRENT_DSF_ROOTS is an empty list. [It follows that, for all voxels x at all times during the execution of the algorithm, x.DSFnode is NIL if, and only if, x has never been an argument of a call of MAKE_SET() or UNITE_SETS().]

After this initialization, the rest of the algorithm is stated by the pseudocode below. The loop on lines 4–7 transforms the DSF from a representation of $C_f(\tau_{i-1})$ to a representation of $C_f(\tau_i)$. [Note that x.DSFnode must be NIL on entry to the inner loop on lines 5–6, because the voxel x will not have been an argument of any earlier call of MAKE_SET() or UNITE_SETS(). So MAKE_SET(x) will be called on line 7 if, and only if, y.DSFnode = NIL for every 6-neighbor y of x.] Lines 8–11 create the vertices of the FHT that have level τ_i, and assign the children to the newly created vertices.

```
1. for $i \leftarrow 1$ to $k$ do
2.     foreach $\nu \in$ CURRENT_DSF_ROOTS do $\nu$.oldVertex $\leftarrow$ $\nu$.vertex;
3.     previousDSFroots $\leftarrow$ a copy of the list CURRENT_DSF_ROOTS;
4.     foreach $x \in B[i]$ do
5.         foreach 6-neighbor $y$ of $x$ in $X$ do
6.             if $y$.DSFnode $\neq$ NIL then UNITE_SETS($x, y$);
7.         if $x$.DSFnode = NIL then MAKE_SET($x$);
8.     foreach $\nu \in$ CURRENT_DSF_ROOTS do
9.         $\nu$.vertex $\leftarrow$ a new FHT vertex $v$ with $L(v) = \tau_i$ and $D(v) = \nu$;
10.    foreach $\nu \in$ previousDSFroots do
11.        Make (DSF_ROOT($\nu$)).vertex the parent of $\nu$.oldVertex;
```

The UNITE_SETS(x, y) operation and the DSF_ROOT(ν) function are implemented using union-by-rank with path-compression [2, p. 447]. The running time of a sequence of m calls of MAKE_SET, UNITE_SETS and DSF_ROOT is $O(m\beta(m))$, where $\beta(m)$ is an *extremely* slowly growing function of m [2, p. 449]. [$\beta(m) = \alpha(m, m)$, where α is an inverse of Ackermann's function.] In fact $\beta(m) = 3$ or 4 for all values of $m \geq 8$ that might occur in any conceivable application. In our applications the number k of levels or bins is very much smaller than the number $|X|$ of voxels in the domain X of the image f, and $\sum_{i=1}^{k} |C_f(\tau_i)|$ is also smaller than $|X|$. In this context the time complexity of the algorithm is $O(|X| \log k + |X|\beta(|X|))$, where the term $|X| \log k$ corresponds to the time complexity of initializing the k bins, and the factor $\log k$ assumes the use of binary search to find the appropriate bin for each voxel. As $\beta(|X|) = 3$ or 4 for all images of practical interest, the time-complexity is "essentially" $O(|X| \log k)$. If the levels τ_i are regularly spaced, then the bin for each voxel can be found in $O(1)$ time and the time-complexity of the algorithm is essentially $O(|X|)$.

3.2 Preliminary Results and the Need for Simplification

We applied our method to several macromolecules, and here we present its application to a reconstruction of the e. coli 70s ribosome [5], and a helical reconstruction of drosophila kinesin dimer AMP-PNP state [6]. Figs. 1 and 2 show surface visualizations and cross-sections of these specimens. Figs. 3(a) and 4(a) show associated FHTs. We used the drawgram program in the package PHYLIP [7] to generate these tree images. Each vertex that has more than one child is represented by a horizontal segment. The presence of a downward segment from a horizontal segment a to a horizontal segment or endpoint d indicates that the vertex represented by d is a descendant of the vertex represented by a, and the length of the downward segment is proportional to the difference between the levels of those vertices. In each case, we eliminated the vertical segment from the root of the tree.

These FHTs are given by the algorithm of Sect. 3.1 with $k = 128$ and, for $1 \leq i \leq 128$, $\tau_i = f_{\max} - i\Delta$, where $\Delta = (f_{\max} - f_{\min})/128$.

While they appear to be different, the trees of Figs. 3(a) and 4(a) are so cluttered that one cannot be sure whether or not the difference is just an artifact of the display. We need to construct simpler FHTs that better reveal the structural essence of the molecules.

3.3 Pruning FHTs by Component Size

Pruning an FHT by component size removes all vertices that correspond to components containing fewer than a certain number of voxels. This transforms an FHT T for an image f to an FHT T' for f such that Vertices$(T') = \{v \in \text{Vertices}(T) \mid |D_T(v)| \geq \delta\}$, in which the functions $L_{T'}$ and $D_{T'}$ are the restrictions to Vertices(T') of L_T and D_T. Here δ is a positive integer parameter that represents the minimum allowed component size. [This operation cannot disconnect the tree, for if a vertex u of an FHT is the parent of a vertex v, then $|D(v)| \geq \delta$ implies $|D(u)| \geq \delta$.]

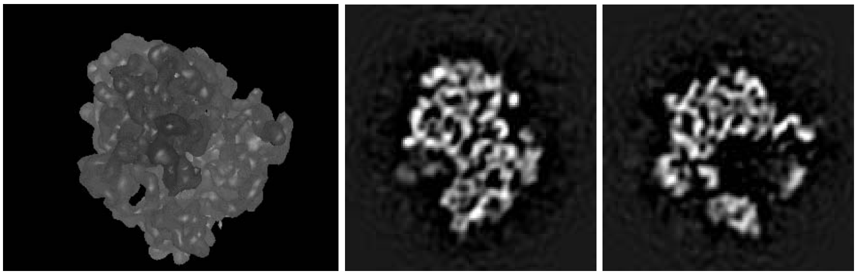


Fig. 1. EMD-1006: E.coli 70s ribosome / ribosome-bound termination factor RF2, surface visualization and cross-sections (51 and 80 of 130)

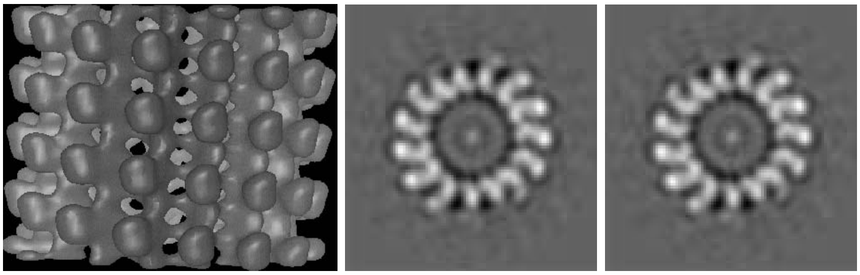


Fig. 2. EMD-1032: Drosophila kinesin dimer AMP-PNP state, surface visualization and cross-sections (47 and 63 of 100)

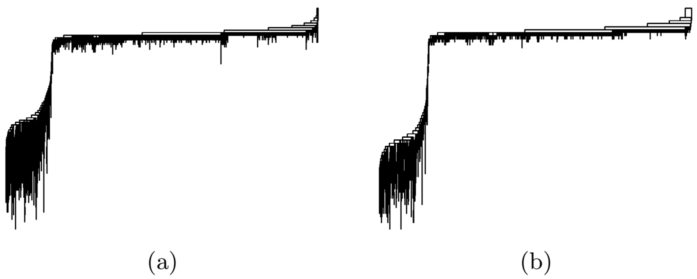


Fig. 3. FHTs based on EMD-1006: (a) unpruned, (b) pruned by minimum component size of 25

It is easy to incorporate pruning by component size into the algorithm of Sect. 3.1 for producing an FHT. The pseudocode need not be changed. It suffices to store the size of the set of voxels that is represented by each DSFtree in a field of its root DSFnode, and include in the list CURRENT_DSF_ROOTS just those root DSFnodes whose size fields are greater than or equal to δ. The algorithm

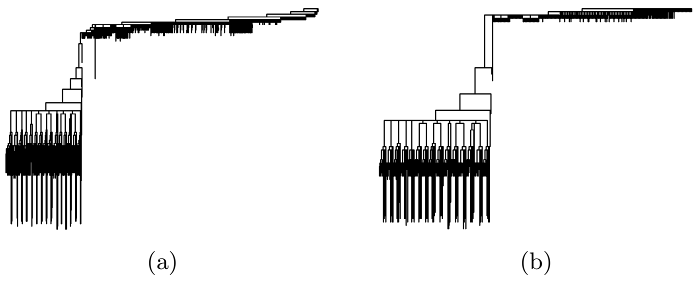


Fig. 4. FHTs based on EMD-1032: (a) unpruned, (b) pruned by minimum component size of 20

will then construct the tree that would be obtained if we pruned the FHT that is produced by the original version of the algorithm.

Figs. 3(b) and 4(b) show pruned FHTs of the images of Figs. 1 and 2.

3.4 Pruning FHTs by Subtree Height

The FHTs that are constructed as described above tend to have "comb-like" structures: There are many leaves near the root. [See the right side of each tree in Figs. 3 and 4.] These leaves correspond to components of the foreground voxel set at very low gray values, and seem mostly to represent components of the ice in which the specimen under study is embedded.

Pruning by subtree height is a second method of pruning that can be applied to FHTs. It eliminates the above-mentioned leaves and some other artifacts that are likely to be due to noise.

Pruning by subtree height h transforms an FHT T for an image f to an FHT T' for f such that $\text{Vertices}(T') = \{v \in \text{Vertices}(T) \mid \text{subtree-height}_T(v) \geq h\}$, in which the functions $L_{T'}$ and $D_{T'}$ are the restrictions to $\text{Vertices}(T')$ of L_T and D_T. Here $\text{subtree-height}_T(v) = \max\{L(w) - L(v) \mid w \text{ is a descendant of } v \text{ in } T\}$. Figs. 5 and 6 show the effects of this operation on the FHTs of Figs. 3(b) and 4(b), for two different values of h.

3.5 Elimination of Short Edges from FHTs

Pruning by subtree height only helps near the leaves. Other parts of the tree will not be simplified unless one chooses a very high value of h to prune by, in which case one could lose much valuable information.

We have developed another method of simplifying FHTs that can be applied after pruning by subtree height, and which does not have this shortcoming. We call this method *elimination of short edges*, because it essentially eliminates edges *anywhere* in the tree that are not longer than a positive parameter ϵ. Figs. 7 and 8 show its effect on the FHTs of Figs. 5(a) and 6(a), for two different values of the parameter ϵ.

Simplification of an FHT T by elimination of short edges can be accomplished by calling $\text{SIMPLIFY}(\text{root}(T), \text{root}(T), \epsilon)$, where $\text{root}(T)$ is the root of T and SIMPLIFY() is defined as follows:

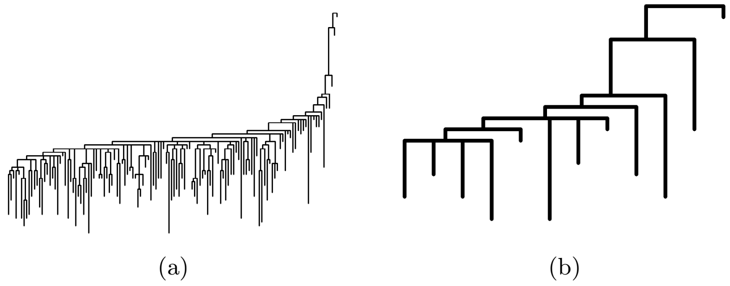


Fig. 5. FHTs based on EMD-1006: minimum component size of 25, pruned by subtree height parameter (a) $h = 4\Delta$, (b) $h = 20\Delta$, where $\Delta = (f_{\max} - f_{\min})/128$

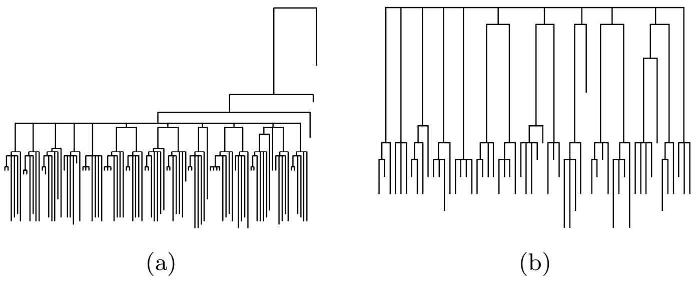


Fig. 6. FHTs based on EMD-1032: minimum component size of 20, pruned by subtree height parameter (a) $h = 4\Delta$, (b) $h = 20\Delta$, where $\Delta = (f_{\max} - f_{\min})/128$

```
SIMPLIFY(Vertex $v$, Vertex $r$, float $\epsilon$):
    **foreach** child $c$ of $v$ **do**
        **if** $L(c) > L(r) + \epsilon$ **then**
            **if** $r \neq v$ **then** make $r$ the parent of $c$;
            SIMPLIFY$(c, c, \epsilon)$;
        **else**
            SIMPLIFY$(c, r, \epsilon)$;
            Remove the vertex $c$;
```

We can give a non-recursive characterization of the effect of this simplification method. Let T be any FHT for an image f. We define an *ϵ-acceptable simplification* of T to be an FHT T' for f that satisfies the following conditions:

1. Vertices$(T') \subseteq$ Vertices(T), and the functions $L_{T'}$ and $D_{T'}$ are the restrictions to Vertices(T') of the functions L_T and D_T.
2. For all $p, c \in$ Vertices(T') such that p is the parent of c in T', $L(c) > L(p) + \epsilon$.

Let $\leq$ denote the partial order, on the set of all ϵ-acceptable simplifications of T, such that $T_1 \leq T_2$ if, and only if, each vertex in Vertices$(T_1) \setminus$ Vertices(T_2) has an ancestor in T that lies in Vertices$(T_2) \setminus$ Vertices(T_1). Then the FHT that is

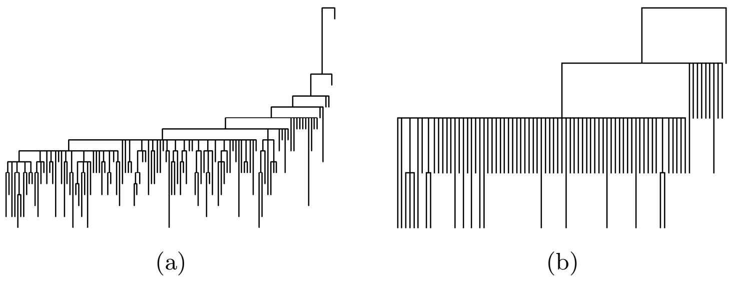


Fig. 7. FHTs based on EMD-1006: minimum component size of 25, pruned by subtree height parameter $h = 4\Delta$ and short edges eliminated by parameter (a) $\epsilon = 2\Delta$, (b) $\epsilon = 9\Delta$, where $\Delta = (f_{\max} - f_{\min})/128$

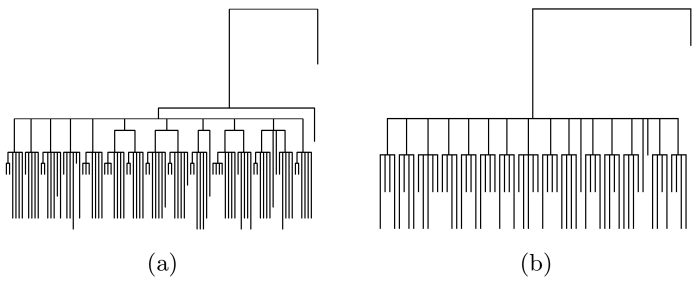


Fig. 8. FHTs based on EMD-1032: minimum component size of 20, pruned by subtree height parameter $h = 4\Delta$ and short edges eliminated by parameter (a) $\epsilon = 2\Delta$, (b) $\epsilon = 9\Delta$, where $\Delta = (f_{\max} - f_{\min})/128$

produced from the FHT T by SIMPLIFY(root(T), root(T), ϵ) is the ϵ-acceptable simplification of T that is maximal with respect to $\leq$.

A consequence of the fact that elimination of short edges affects all parts of the tree is that, for similar values of the parameters ϵ and h, elimination of short edges will typically remove many more vertices than pruning by subtree height.

4 Discussion

EM reconstruction need not preserve the geometry of the specimens under study. The foreground history tree (FHT) of a 3D gray-valued voxel-based image is a useful descriptor that is insensitive to topology-preserving transformations.

We have presented parametric methods of simplifying FHTs that remove artifacts due to noise, while preserving what seem to be essential spatial properties of the specimens. The simplified FHTs of different specimens obtained from a molecular database capture some of the structure in those specimens. For example, the FHTs in Fig. 8 of the helical structure shown in Fig. 2 contain a vertex from which many similar-looking subtrees are descended.

FHTs provide a potentially useful way of discretely representing essential aspects of the shape of a complicated object. Thus we believe that they can be used in methods of querying macromolecular databases.

Acknowledgments

This work was supported by the National Institutes of Health Grant HL070472. Our interactions with P. L. Combettes, J.-M. Carazo, R. Marabini, E. Garduño, S. H. W. Scheres, and I. Montealegre were helpful in preparing this manuscript.

References

1. Rosenfeld, A.: Three-dimensional digital topology. Information and Control **50** (1981) 119–127
2. Cormen, T.H., Leiserson, C.E., Rivest, R.L.: Introduction to Algorithms. MIT Press, Cambridge, MA, USA (1990)
3. Carr, H., Snoeyink, J., Axen, U.: Computing contour trees in all dimensions. Computational Geometry: Theory and Applications **24** (2003) 75–94
4. Rosenfeld, A., Kak, A.C.: Digital Picture Processing. Academic Press, New York, NY, USA (1976)
5. Rawat, U., Zavialov, A.V., Sengupta, J., Valle, M., Grassucci, R.A., Linde, J., Vestergaard, B., Ehrenberg, M., Frank, J.: A cryo-electron microscopic study of ribosome-bound termination factor RF2. Nature **421** (2003) 87–90
6. Hoenger, A.: A new look at the microtubule binding patterns of dimeric kinesins. Journal of Molecular Biology **297** (2000) 1087–1103
7. Felsenstein, J.: PHYLIP - phylogeny inference package (version 3.2). Cladistics **5** (1989) 164–166

Fusing Features in Direct Volume Rendered Images

Yingcai Wu, Huamin Qu, Hong Zhou, and Ming-Yuen Chan

Department of Computer Science and Engineering
Hong Kong University of Science and Technology
{wuyc, huamin, zhouhong, pazuchan}@cs.ust.hk

Abstract. In this paper, we propose a novel framework which can fuse multiple user selected features in different direct volume rendered images into a comprehensive image according to users' preference. The framework relies on three techniques, i.e., *user voting*, *genetic algorithm*, and *image similarity*. In this framework, we transform the fusing problem to an optimization problem with a novel energy function which is based on user voting and image similarity. The optimization problem can then be solved by the genetic algorithm. Experimental results on some real volume data demonstrate the effectiveness of our framework.

1 Introduction

Direct volume rendering (DVR) is a powerful and flexible volume visualization tool and has been widely used in many fields. However, to effectively and intuitively explore volumetric data through DVR still remains a challenging issue. Due to the data occlusion and human perception, one of the difficulties is to develop effective visualization techniques capable of revealing multiple features simultaneously. To address this issue, many approaches like illustrative visualization and importance-based techniques have been proposed. However, most techniques have more or less limitations. On the other hand, as it is easier for users to reveal a specific feature than to highlight multiple features simultaneously in a *direct volume rendered image* (DVRI), a feasible solution to the problem may allow users to select multiple features in distinct DVRIs, and then automatically fuse those features into a comprehensive DVRI. Therefore, users can focus on one feature each time and the task for exploring multiple features is simplified.

We assume that a series of DVRIs have been generated and cached along with the *transfer functions* (TFs) used. To fuse multiple features in different DVRIs, there are two straightforward solutions, i.e., to blend those images, or to generate a new DVRI with a new TF created by linearly combining the previous TFs. However, they both fail to achieve our goal in most cases. Image blending does not work for our purpose due to its limitations like losing depth cues and introducing misleading information [1]. Linear combination of the TFs does not work either. This can be mainly attributed to the nonlinear operations of the integration used in DVR. For example, given two DVRIs and their corresponding TFs (see Figure 2(a) and 2(b)), if users want to reveal features appearing in both 2(a) and 2(b) by linear combination of the TFs, they can only obtain something like 2(c) no matter what α and β are, which is far from what they expect such as 2(d). Because there are other features, such as the intestines and muscles between intensity d_1 and d_2 , linear combination in this example does not work.

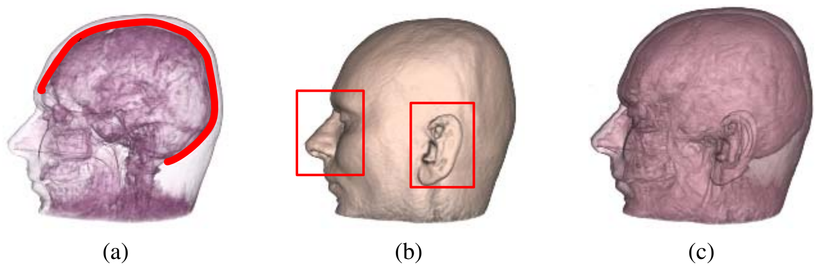


Fig. 1. Experiment on an MRI human head dataset: (a)-(b) Parent images with user selected features; (c) Child image generated by fusing the user selected features shown in (a) and (b)

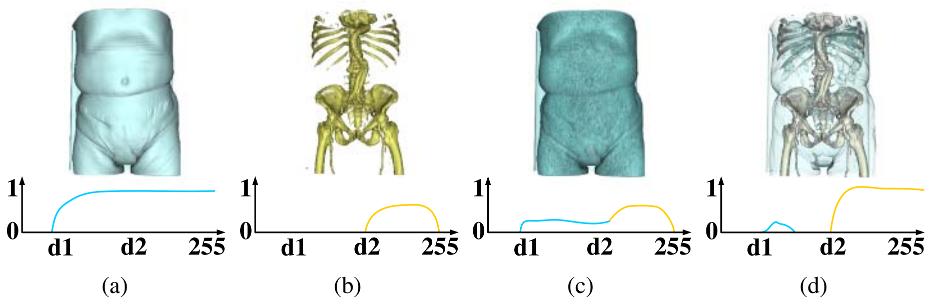


Fig. 2. (a)-(b) Parent images 1 and 2, and their TFs TF_1 and TF_2; (c) Child image rendered with linearly combined TF : $TF3 = \alpha \times TF1 + \beta \times TF2$, where $\alpha = 0.3$ and $\beta = 1$; (d) Child image rendered with our method by fusing (a) and (b)

To solve the general feature fusing problem, we propose a novel fusing framework, which relies on three techniques, i.e., *user voting*, *genetic algorithm (GA)*, and *image similarity*. In this framework, we transform the fusing problem to an optimization problem with a novel energy function based on user voting and image similarity, which can then be solved by the GA. User votes includes the user selected features in DVRIs and the corresponding user given scores representing how much the users like the features. A selected feature could be either a region of interest (with high scores) or the context (with low scores). Our approach contains some advanced characteristics. First, as it is easier for users to select features in DVRIs than in volume slices [2], our approach allows users to select multiple features directly in DVRIs instead of in volume slices. The features can be then automatically fused together. Second, we develop a general framework for fusing multiple user-selected features in DVRIs. Compared with previous methods such as image blending and TF interpolation, our method is more robust and general. Third, as a semi-automatic method, our approach integrates user knowledge into the fusing process. Since the visualization goal highly depends on the tasks and users, our system can gain valuable input from users by user voting.

This paper is organized as follows: After reviewing previous work in Section 2, we give an overview of our system in Section 3. The solution encoder/decoder is explained in Section 4. A genetic algorithm used to solve the optimization problem is described in Section 5. In Section 6, we provide the details of the image similarity metric. Experimental results and discussions are presented in Section 7. We conclude and suggest some future work in Section 8.

2 Related Work

Feature-based Visualization. Weiskopf et al. [3] proposed several interactive clipping methods by exploiting the powerful capability of GPU. Viola et al. [4] developed a novel importance-based approach capable of enhancing important features while retaining necessary context by generating cut-away and ghosted images from volumetric data. Volume illustration techniques, first introduced by Ebert et al. [5], provide an alternative way for focus + context visualization with abstraction techniques (non-photorealistic rendering). Wang et al. [6] presented an interactive GPU-assisted volume lens to magnify the regions of interest while preserving the context by compressing the other volume regions.

Transfer Function (TF) Design. An excellent survey on TF design can be found in [7]. Marks et al. [8] developed an image-centric system which automatically generates many perceptually different images and organizes them efficiently for users to select. Kindlemann and Durkin [9] presented a histogram data structure used to semi-automatically obtain a good TF with users' guidance. The images generated by the both approaches can be used as inputs to our system. Ma [10] proposed a novel approach based on image graphs to facilitate visual data exploration. König and Gröller [11] developed a TF design interface paradigm which provides several specification tools for each search domain. Although Ma's and König et al.'s methods used linear combination of TFs, they cannot always get the expected results (see Figure 2).

Genetic Algorithms. Genetic algorithms are widely used in many fields like computer graphics [12]. He et al. [13] first employed GAs to generate TFs. Our approach is inspired by their work, but aims at fusing different features rather than TF design from scratch. In addition, compared with He et al.'s approach, our approach is based on image similarity and user knowledge (or user voting). Users are allowed to control which features will be retained or enhanced in the child images by user voting. House et al. [14] used a GA to choose visualization parameters to optimize visualization quality.

3 System Overview

To simplify the presentation, we first introduce the following terms: *Parent image* for the image with user selected features for fusing; *Child image* for the fused image from parent images; *Parent TF* for the TF corresponding to the parent image; *Child TF* for the TF corresponding to the child image. All the images are rendered by DVR.

Our system consists of four components as shown in Figure 3, i.e., *solution encoder/decoder*, *genetic algorithm solver*, *direct volume renderer*, and *image similarity evaluator*. The encoder/decoder component specifies the genome representation by analyzing the parent transfer functions (TFs) (denoted as $TF_1 \cdots TF_n$ in Fig. 3), and then

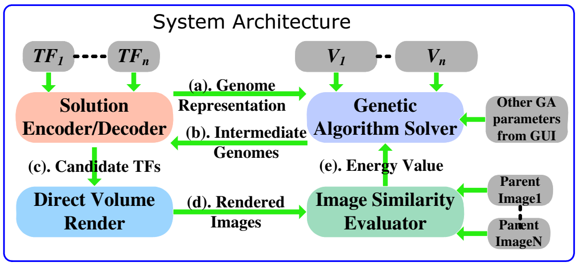


Fig. 3. System Architecture

sends the genome representation to the genetic algorithm (GA) solver (see Figure 3 (a)). The genetic algorithm solver generates new genomes (the encoded TFs) by imitating the process of natural evolution. The expected similarity values denoted as $V_1 \cdots V_n$ in Fig. 3 and the calculated similarity values received from the image similarity evaluator for each parent image are used to form an energy function which measures the differences between the real calculated similarity values and the expected similarity values. The smaller the energy value, the fitter the corresponding genome. Each intermediate genome created in the course of the GA evolution is then sent back to the solution encoder/decoder (see Figure 3(b)). The decoder converts the genome to a candidate TF which is then passed to the direct volume renderer (see Figure 3(c)). The direct volume renderer generates a child image using DVR techniques for the candidate TF. The generated image is then passed to the image similarity evaluator (see Figure 3(d)). It measures the similarity between the child image and each of the parent images. The image similarity values are then sent back to the genetic algorithm solver (see Figure 3(e)) to begin the next cycle of refinement.

4 Solution Encoder/Decoder

The solution encoder/decoder specifies the genome representation by analyzing the parent TFs. To define the optimal genome representations [15], our approach represents a TF as a one dimensional (1D) array of floating numbers. The component first smoothens the parent TFs using a gaussian function to filter out the high frequencies to obtain bandlimited signals. After that, it samples TFs adaptively above the Nyquist frequency. The samples are then used to specify the genome representation. In addition, they can be used to restrict the search space to improve the GA performance. For example, in Figure 4 there are two parent TFs denoted by different line patterns to be fused. The points on the axis of the scalar value are the union of the sampling positions of the parent TFs. The vertical dashed lines start with 0 and end with the maximum opacity value of the parent TFs. They act as the range of the opacity values of the corresponding points on the axis of the scalar value. All the opacity values on the forementioned points (on the axis of the scalar value) of each candidate child TF constitute a genome.

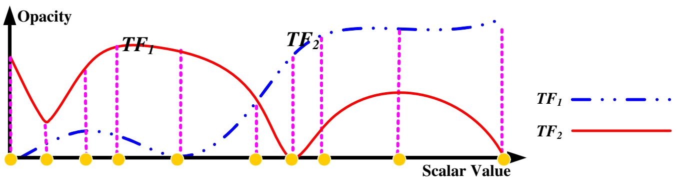


Fig. 4. The genome representation (the points on the axis of the scalar value) obtained by analyzing parent transfer functions TF_1 and TF_2

5 Genetic Algorithm

A *Genetic Algorithm* (GA) is a search algorithm imitating the process of natural evolution [15]. It is particularly useful for searching solutions to optimization problems, especially when the search space is huge and unknown. The pseudo-code for the simple genetic algorithm used in our system is shown as follows:

```
1: Randomly create an initial population of $n$ genomes (encoded TFs)
2: repeat
3:    repeat
4:       Select a pair of genomes from the current population using roulette wheel scheme as
         parent genomes such that fitter (or better) genomes are more likely to be chosen
5:       Crossover (two-point crossover) the selected pair with probability $p_r$ or exactly copy
         (or clone) the pair with probability $1 - p_r$ to form two new genomes
6:       Mutate the two newly created genomes with mutation probability $p_m$
7:    until $n$ new genomes (offsprings) have been created
8:    Replace the current population with the $n$ new genomes
9: until Terminating conditions such as the converging of the GA are met
```

Energy functions are used to measure the fitness of genomes, and return the measurement to the GA where the energy values are used to determine which genomes in the current population are more likely to be selected to survive. Thus, the energy function has a great impact on the performance of GAs. For our system, we exploit an energy function based on image similarity and user voting to objectively evaluate the fitness of genomes as follows:

$$F = \sum_{k=1}^{n} V_k * |V_k - S_k| \tag{1}$$

Where n is the number of parent images, and V_k represents the vote (or the scores) given by users for the features in parent image k, and S_k denotes the computed image similarity value between the child image and parent image k. The more detailed definition of S_k can be found in Equations (2) in Section 6.

The V_k, from another point of view, can be also considered as the similarity value expected by users between the child image and parent image k. It ranges from 0 to 1. It penalizes the difference between the computed similarity S_k and the user-voted (or user-expected similarity) value V_k. It can guarantee that the child image will get proportional contributions from all the parent images and will not be overwhelmed by any single one. The GA used in our system views the genomes with smaller energy values as the fitter ones for the problem. Using the energy function, we are able to transform the feature fusing problem to an optimization problem, i.e., minimizing the energy function.

6 Image Similarity

In our system, we employ a contour-based similarity metric to compare two DVRIs. The contour is one of the most important perception clues to 3D structures and can be used to compute the similarity of objects. Thus, our system first converts the DVRIs to grey-scale images, and detects the edge images from the grey-scale images using the *Canny edge detector*. Because most contours appearing in the DVRIs also appear in the grey-scale images, our contour-based metric still works well. After that, a Gaussian filter is applied to smooth the edge images so that the pixels without an edge covering can also obtain contributions from the nearby edges. We set the size of the filter to be 5×5 for 512×512 images, and 3×3 for 256×256 images. The image similarity can then be computed pixel by pixel.

The image similarity value S_k between the edge image of the child image and the edge image of each parent image k is computed as follows:

$$S_k = \frac{\sum_{y=1}^{height} \sum_{x=1}^{width} Q(x,y)}{N_{parent}} \tag{2}$$

$$Q(x,y) = \begin{cases} 1 \text{ if } P(x,y) < threshold \\ 0 \quad Otherwise \end{cases} \tag{3}$$

$$P(x,y) = |K_{(x,y)} - K'_{(x,y)}| \tag{4}$$

where N_{parent} in Equation (2) is the number of all pixels on the edges of the parent k's edge image. $threshold$ in Equation (3) is a parameter set by the system, and $P(x,y)$ in Equation (4) represents the difference between two corresponding pixels and K and K' are the Gaussian filtered child and parent edge images with resolution $(width, height)$.

Notice that we consider only the pixels on the parent edge image k for S_k, i.e., Equation (2) is only computed if $K'_{(x,y)} \neq 0$. If there are user-selected features in the DVRIs which are to be compared, our system considers only the pixels within these features. Thus, the N_{parent} in Equation (2) becomes the number of pixels within the features on the edges of the parent edge image.

To determine the default $threshold$ in Equation (3), we conducted extensive experiments on real volume data. All the edge images to be compared are 8-bit grey-scale images. After extensive experiments, we observed that the similarity values returned by the similarity evaluator was very similar to what users perceived when the $threshold$ ranged from 60 to 90. Based on this, the $threshold$ was fixed to be 80 for all experiments in this paper. An example of computed tomography (CT) human head volume data is shown in Figure 5. Figure 5(a) and 5(c) are two parent images. Figure 5(e) was generated by fusing features appearing in Figure 5(a) and 5(c). Figure 5(b), 5(d), and 5(f) are their corresponding edge images. Figure 5(g) shows the similarity value distribution in terms of different $threshold$. From the figure, we can observe that when the $threshold$

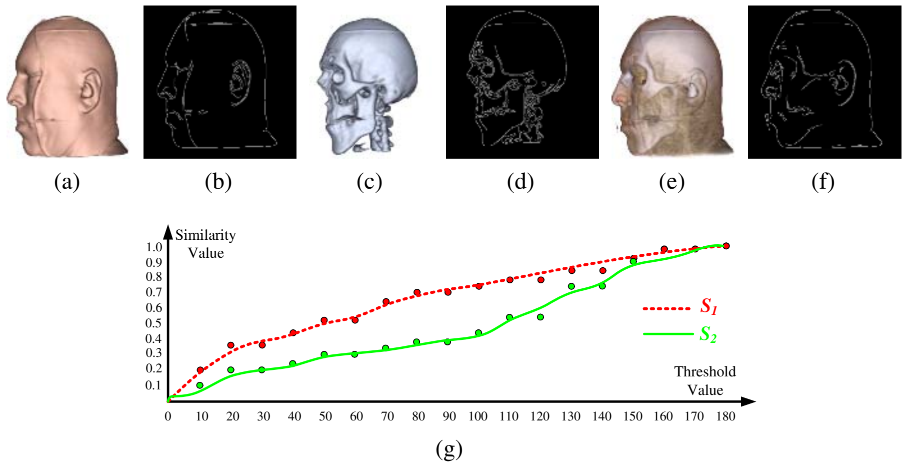


Fig. 5. Experiments for image similarity on CT human head volume data with different *threshold* in Equation 3: (a) Parent image 1; (b) Parent edge image 1; (c) Parent image 2; (d) Parent edge image 2; (e) Child image; (f) Child edge image; (g) Similarity value distribution in terms of different *threshold*

ranges from 60 to 90, the similarity value S_1 for Figure 5(e) and 5(a) will vary from 0.53 to 0.7, while S_2 for Figure 5(e) and 5(c) will vary from 0.3 to 0.38. This is similar to what users perceive. We also carried out the experiments on other volumetric data, such as CT engine data, CT human tooth data, and Magnetic resonance imaging (MRI) head data and found similar results. The image resolution was 512×512 and the size of the gaussian filter we applied was 5×5. Finally, we did the above experiments on images with 256×256 image resolution and with a 3×3 gaussian filter and obtained similar findings as well. Thus, to improve the system's performance, we can use the 256×256 image resolution and 3×3 gaussian filter in the fusing process for DVR and image similarity computing.

7 Experiment Results and Discussions

Our system was implemented in C++ based on the VTK 5.0 library [16]. We tested the system on a Pentium(R) 4 3.2GHz PC with 1GB RAM and an Nvidia Geforce 6800 Ultra GPU with 256MB RAM. The sampling rate of DVR was two samples per voxel along each ray. For the sake of performance, the rendered image resolution was 256×256 in the fusing process and was then switched to 512×512 after fusing. Following Jong's suggestion [17], we set the population size to be 10, the crossover rate to be 0.6, the mutation rate to be 0.05, n to be 20, and k to be 1.005 for the GA. In the following experiments, we obtained sufficiently good results within acceptable time frames.

We carried out the first experiment on a CT engine dataset ($256 \times 256 \times 128$) to verify the effectiveness of the fusing while taking user-selected features into account. We gave 0.4 scores to all the features shown in Figure 6(a) and 0.6 scores to the feature selected by a rectangle in Figure 6(b). The result is shown in Figure 6(c). From the result, we can

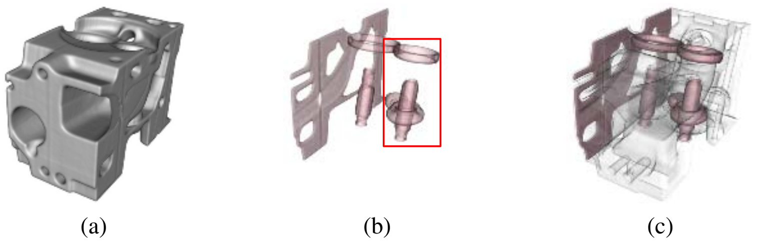


Fig. 6. Experiment on a CT engine dataset: (a) Parent image 1; (b) Parent image 2 with a feature selected with a rectangle; (c) Child image generated with $V_1 = 0.4$ and $V_2 = 0.6$

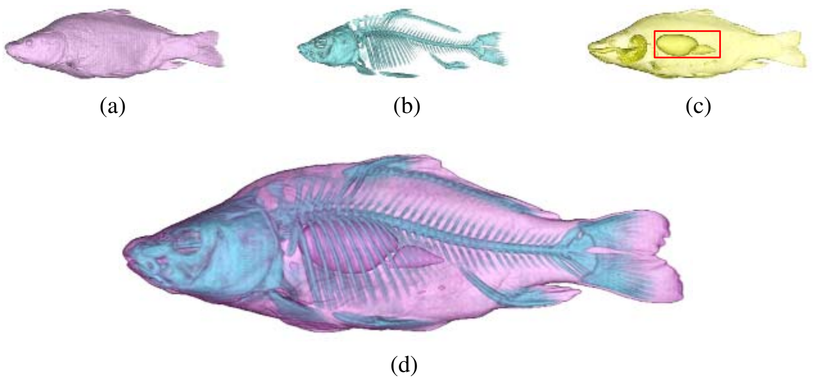


Fig. 7. Experiment on a CT carp dataset: (a) Parent image 1; (b) Parent image 2; (c) Parent image 3 with a user-selected feature indicated by a rectangle; (d) Child image obtained by fusing features in (a) with 0.3 scores, features in (b) with 0.4 scores, and the user-selected feature in (c) with 0.3 scores

see that the user-selected feature is emphasized in the child image according to users' preference. The result was generated within 48 seconds.

Our approach was not limited to fusing only two DVRIs. In the second experiment, we fused the features in multiple DVRIs into a comprehensive one based on users' preference. We carried out the experiment on a CT carp dataset ($256 \times 256 \times 512$). In this experiment, we fused all the features shown in Figure 7(a) with 0.3 scores and 7(b) with 0.4 scores, and a feature (the swimming bladder) selected by a rectangle in Figure 7(c) with 0.3 scores together into a comprehensive image. Figure 7(d) shows the resulting image generated within 40 seconds.

We did the last experiment on an MRI head dataset ($256 \times 256 \times 256$) to demonstrate that our approach is able to generate a fused image containing the user selected features with clearer contours. The image shown in Figure 1(b) was created by a semi-automatic TF design method [9]. However, the semi-automatic method could not easily obtain an image revealing the inner structures of the head. Thus, we manually created an image (see Figure 1(a)) with a simple TF capable of revealing the inner structures of the data. But this image was not as good as what we expected, because the ear disappeared and the inner structures like the brain were not clear enough. To solve these problems, we specified the features with curves in Figure 1(a) and with rectangles in 1(b). After that, we fused these features with $V_1 = 0.6$ and $V_2 = 0.4$. In the fusing process, our approach favored only those candidate images with clear contours within the specified regions. Figure 1(c) shows the result generated within 47 seconds.

8 Conclusion and Future Work

In this paper, we present a novel visualization technique which fuses multiple user selected features in distinct DVRIs into a comprehensive DVRI. We develop a general framework for the fusing problem. We view the fusing problem as a parameter optimization problem, which can then be solved using a genetic algorithm. To measure the fitness of a candidate image, we propose an energy function based on image similarity and user voting. Our approach is easy to use, especially for non-experts like physicians, because it is much easier for them to extract one feature in the volume data than to reveal multiple features simultaneously. In addition, our system allows users to directly select and identify features in DVRIs, which is more convenient and accurate than in volume slices. To fuse multiple features from several DVRIs, users usually do not need to set any parameters other than the scores voted for the features.

There are many possible venues for future work. Our implementation was not highly optimized for performance. The running time for each fusing on the tested volumetric datasets was about 50 seconds. We plan to exploit GPU-accelerated GAs [18] and DVR to greatly reduce the running time of our algorithm. We also plan to develop more sophisticated image similarity metrics for complex transparent DVRIs and test them with more real volume datasets. We assume that the viewpoint stays unchanged in the fusing process. We will extend our system to fuse features in DVRIs rendered from different viewpoints.

Acknowledgements

This work was supported by RGC grant CERG 618705 and HKUST grant DAG 04/05 EG02. We would like to thank the anonymous reviewers for their helpful comments.

References

1. Baudisch, P., Gutwin, C.: Multiblending: displaying overlapping windows simultaneously without the drawbacks of alpha blending. In: Proceedings of the SIGCHI conference on Human factors in computing systems. (2004) 367–374
2. Tzeng, F.Y., Lum, E.B., Ma, K.L.: An intelligent system approach to higher-dimensional classification of volume data. IEEE Transactions on Visualization and Computer Graphics **11** (2005) 273–284
3. Weiskopf, D., Engel, K., Ertl, T.: Interactive clipping techniques for texture-based volume visualization and volume shading. IEEE Transactions on Visualization and Computer Graphics **9** (2003) 298–312

4. Viola, I., Kanitsar, A., Gröller, M.E.: Importance-driven feature enhancement in volume visualization. IEEE Transactions on Visualization and Computer Graphics **11** (2005) 408–418
5. Rheingans, P., Ebert, D.: Volume illustration: Nonphotorealistic rendering of volume models. IEEE Transactions on Visualization and Computer Graphics **7** (2001) 253–264
6. Wang, L., Zhao, Y., Mueller, K., Kaufman, A.E.: The magic volume lens: An interactive focus+context technique for volume rendering. In: Proceedings of IEEE Visualization. (2005) 367–374
7. Kindlmann, G.: Transfer functions in direct volume rendering: Design, interface, interaction. In: Course notes of ACM SIGGRAPH. (2002)
8. Marks, J., Andalman, B., Beardsley, P.A., Freeman, W., Gibson, S., Hodgins, J., Kang, T., Mirtich, B., Pfister, H., Ruml, W., Ryall, K., Seims, J., Shieber, S.: Design galleries: a general approach to setting parameters for computer graphics and animation. In: Proceedings of ACM SIGGRAPH. (1997) 389–400
9. Kindlmann, G., Durkin, J.W.: Semi-automatic generation of transfer functions for direct volume rendering. In: Proceedings of Volume Visualization Symposium. (1998) 79–86
10. Ma, K.L.: Image graps - a novel interface for visual data exploration. In: Proceedings of IEEE Visualization. (1999) 81–88
11. König, A.H., Gröller, E.M.: Mastering transfer function specification by using volumepro technology. In: Proceedings of Spring Conference on Computer Graphics. (2001) 279–286
12. Sims, K.: Artificial evolution for computer graphics. In: Proceedings of ACM SIGGRAPH. (1991) 319–328
13. He, T., Hong, L., Kaufman, A., Pfister, H.: Generation of transfer functions with stochastic search techniques. In: Proceedings of IEEE Visualization. (1996) 227–234
14. House, D., Bair, A., Ware, C.: On the optimization of visualizations of complex phenomena. In: Proceedings of IEEE Visualization. (2005) 87–94
15. Mitchell, M. In: An Introduction to Genetic Algorithms. MIT Press (1996)
16. Kitware: (The visualization toolkit. www.vtk.org)
17. Jong, K.A.D.: An analysis of the behavior of a class of genetic adaptive systems. PhD thesis (1975)
18. Wong, M.L., Wong, T.T., Fok, K.L.: Parallel evolutionary algorithms on graphics processing unit. In: Proceedings of IEEE Congress on Evolutionary Computation. (2005) 2286–2293

Binocular Uncalibrated Photometric Stereo

Hui Kong[1], Pengfei Xu[1], and Eam Khwang Teoh[2]

[1] Panasonic Singapore Labs
Blk 1022 Tai Seng Ave., Singapore 534415
{hui.kong, pengfei.xu}@sg.panasonic.com
[2] School of Electrical and Electronic Engineering
Nanyang Technological University, Singapore 639798
eekteoh@ntu.edu.sg

Abstract. In Uncalibrated Photometric Stereo (UPS), the surface normals and light sources are determined up to a group of ambiguous Generalized Bas-Relief (GBR) transformations. However, it has been shown by previous works to be rather troublesome to solve these ambiguities. In this paper, a framework of Binocular Uncalibrated Photometric Stereo (B-UPS) is given for accurate stereo matching for lambertian and non-lambertian objects. It is also shown that the problem of 3D reconstruction with UPS is converted into that of stereo matching with B-UPS. By this conversion, the intractable GBR transformations can be bypassed. In B-UPS, the Orientation-Consistency cue (OC) [1] for distant-lighting condition and Local-Orientation-Consistency (LOC) cue for non-distant lighting condition are used together for stereo matching, where the combination of both cues is made possible by a planar-area detection method based on a pseudo-normal-map segmentation scheme. Excellent matching and reconstruction results for objects with constant and spatial-varying BRDF demonstrate the superiority of B-UPS.

1 Introduction

In Uncalibrated Photometric Stereo (UPS), the illumination condition is unknown, where addressing the bilinear problem is inevitable. This involves how to remove the inherent ambiguity existing in UPS. The ambiguity problem can be described briefly as follows: Given the input data $\mathbf{I} = [I_1 I_2 ... I_M]$, i.e., M images under different illumination conditions, its factorization into the psudonormals B^* and the psudolights S^* gives the true normals B and the true lights S up to an arbitrary unknown invertible transformation matrix A, $A \in GL(3) : B = B^* A, S = A^{-1} S^*$ because $\mathbf{I} = BS = B^* A A^{-1} S^* = B^* S^*$.

The ambiguity can be removed or reduced by knowing additional information about lights or normals. Hayakawa [2] assumes that at least six light sources are of equal (or known relatively) intensity, or that albedo is uniform (or known up to a global scalar) for at least six normals at a curved surface, this reduce the original ambiguity from the $GL(3)$ group into the group of scaled orthogonal transformation $A = \lambda O (O \in O(3), \lambda \neq 0)$. Integrability is another constraint that requires the normals recovered by photometric stereo to correspond to a continuous surface [3,4]. As shown by Belhumeur et al. [4], integrability constraint can determine the shape and albedo up to a Generalized Bas-Relief ambiguity (GBR). Importantly, the integrability and equal intensity constraints combined reduce the ambiguity to binary convex/concave ambiguity. Recently, the specular reflection component has also been utilized to reduce the original ambiguities in UPS. Drbohlav [5] employs the constraint that all lights reflected around corresponding specular normals must give the same vector (the viewing direction). This assumption reduces the original ambiguity into a 2dof group of transformations. The similar case appears in [6] where the same assumption is taken and it is identified that two specular pixels can resolve the GBR ambiguity. Georghiades [7] incorporates the Torrance-Sparrow reflectance model to resolve the GBR ambiguity by solving an optimization problem. More recently, the example-based photometric stereo approach [1] was proposed. In this method, it is assumed that one or more example objects with similar materials and known geometry are imaged under the same illumination conditions. This approach is based on the *orientation-consistency* cue which states that, under orthographic projection and distant lighting, two surface points with the same surface normal and material exhibit the same radiance.

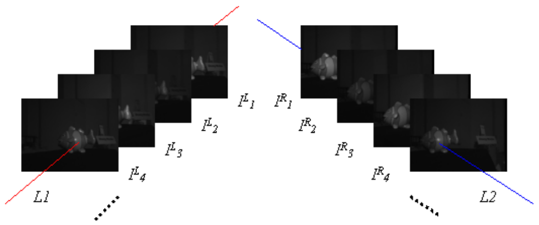


Fig. 1. Two sets of images of a static object taken by a parallel binocular stereo camera with arbitrary lighting: I^L and I^R

In this paper, a framework of Binocular Uncalibrated Photometric Stereo is given for accurate 3D object reconstruction. In this framework, a binocular stereo camera is used in place of the single camera in the conventional UPS. Therefore, two sets of images are obtained by capturing a stationary object with the fixed stereo camera under arbitrary lighting conditions. One is from the left camera and the other is from the right camera. Although either of them can be handled with a specific UPS algorithm, how to utilize the information extracted from the two sets of images, however, has not been investigated so far. The scheme proposed in this paper is different from any of the previous UPS algorithms because of the following merits: (1) Without the need of exemplar objects, B-UPS can also utilized the *orientation-consistency* cue [1] by the introduction of the stereo camera and the problem of 3D reconstruction with UPS is converted into that of stereo matching with the *orientation-consistency* cue. By this conversion, the intractable ambiguity transformations in conventional UPS can be bypassed.

(2) However, if there exist at least two surface points whose normals and albedo are the same (e.g., a plane with uniform texture), the reconstruction

will fail if solely utilizing the *orientation-consistency* cue. To overcome this problem, a *local-orientation-consistency* cue is introduced. The *local-orientation-consistency* cue is based on the non-distant lighting constraint which states that when the light source is close to the target object, the *orientation-consistency* cue is not valid for the whole target object surface but still works for the small local areas of the object surface. Based on the *orientation-consistency* and *local-orientation-consistency* cues, we can capture two groups of images, one is taken under the distant lighting and the other corresponds to the non-distant lighting. These two groups of images are called distant-light and non-distant-light images.

(3) A small planar-area detection algorithm is given based on the pseudo-normal map segmentation. For those non-planar areas, all the images (distant-lighting and non-distant-lighting images) are utilized based on the *orientation-consistency* and *local-orientation-consistency* cues. For those planar areas, the non-distant-lighting images are used based on *local-orientation-consistency* cue.

(4) For non-Lambertian reflectance, e.g., Torrance-Sparrow model, we will show that when the camera is located far enough from the target object, the *orientation-consistency* and *local-orientation-consistency* cues can also be well approximated.

2 Binocular Uncalibrated Photometric Stereo

Let $I_L = \{I_L^1, I_L^2, ..., I_L^M\}$ and $I_R = \{I_R^1, I_R^2, ..., I_R^M\}$ be two sets of images of a static object taken by a fixed binocular stereo cameras when the light source (distant or non-distant light) changes arbitrarily M positions, see Fig.1. The following two parts give a detailed description on how to utilize the *orientation-consistency* cue in B-UPS for reconstruction and the problems in using this cue.

Refer to Fig.1, the red line $L1$ crosses the $[I_L^1, I_L^2, ..., I_L^M]$ at the same image location, p^L. Denote the values of these pixels at p^L as $[p_L^1, p_L^2, ..., p_L^M]$. Correspondingly, the blue line $L2$ intersects $[I_R^1, I_R^2, ..., I_R^M]$ at another location, p^R. Denote the values of these pixels at p^R as $[p_R^1, p_R^2, ..., p_R^M]$. With the assumption of lambertian reflectance and distant lighting (point light source), if p^L and p^R correspond to the same surface point of the target object, and there exists no noise in the two sets of images, $[p_L^1, p_L^2, ..., p_L^M]$ is equal to $[p_R^1, p_R^2, ..., p_R^M]$. Therefore, it is not difficult to find the correspondence between the two views if the normals of any two surface points of the target object are different. However, the *orientation-consistency* cue is valid enough for 3D reconstruction with B-UPS only when the light is assumed to be distant and the target object is lambertian and convex. However, it is unrealistic to reconstruct the real-world object with only such a cue. Whether this cue is enough if the lighting is distant and meantime the object is lambertian and non-convex (e.g., part of the target object surface is a plane with uniform albedo)? Whether this cue is enough if the lighting is non-distant and the object is lambertian and convex? Whether this cue is enough if the lighting is non-distant and the object is lambertian and non-convex? etc. The answers to these questions constitute the major motivation of this paper. We assume the stereo camera to be located far enough from the target object.

In this part, we give a thorough analysis on the *orientation-consistency* cue. Specifically, we will analyze the situations where the *orientation-consistency* cue will fail in reconstruction with B-UPS and where the *orientation-consistency* cue can be used with the other cue for complete reconstruction.

Case 1: Lambertian Reflectance and Convex Object. In this case, when the lighting is far enough that it can be regarded as a point light source, it has been shown by previous analysis that the *orientation-consistency* cue is enough for reconstruction with B-UPS. When the lighting is non-distant, two circumstances need further consideration, i.e., whether the light can or cannot be regarded as point light source. With reference to Fig.2, the light source $\mathbf{s}$ changes arbitrarily M positions closely to the target object from $\mathbf{s}_1$ to $\mathbf{s}_M$. Under this circumstance that the light source is small enough to be regarded as a point, as in Fig.2.(a), no matter whether the albedo is uniform or not, it can shown that the *orientation-consistency* is generally enough for reconstruction with B-UPS. The reason lies as follows: Although there always exist two surface points with normals $\mathbf{n}_1$, $\mathbf{n}_2$ and equal pixel values (if equal albedo) when the light source moves to $\mathbf{s}_i$, however, as long as the light source does not move in the direction of blue line (the bisector of the angle formed by the two red line), their pixel intensities are not equal generally under the other lighting positions (e.g., the light moves to the position of $\mathbf{s}_k$). As for object with non-uniform albedo, the possibility of their pixel intensities being equal at all the lighting positions are even smaller.

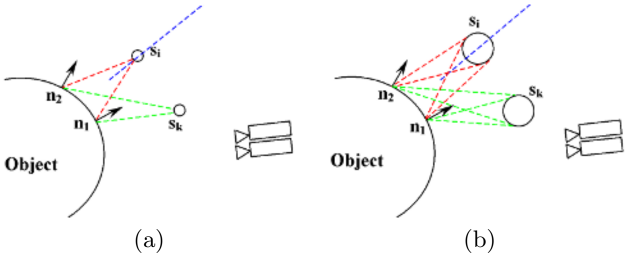
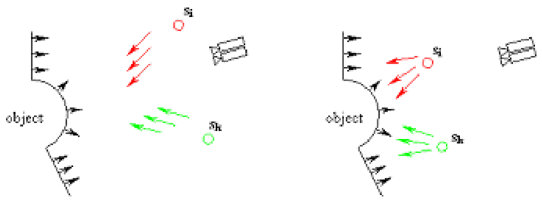


Fig. 2. The illustration of Case 1. In (a), the light source is regarded as point light source whereas, in (b), it is non-point light source.

If the light source cannot be regarded as a point light (e.g., area light source), as in Fig.2.(b), it can be regarded as a cluster of many point light sources. The radiance of each surface point is determined by each of these point light sources (the scaled linear summation of irradiance from each point light source). Similar to the case in Fig.2.(a), as long as the light source does not move along the blue line, the *orientation-consistency* is enough for reconstruction. Since the lighting position is changed arbitrarily instead of along straight line, from Fig.2 (a) and (b), we can generally conclude that the *orientation-consistency* cue is enough for the case when the reflectance of the convex object surface is lambertian.

Case 2: Lambertian Reflectance and Non-Convex Object. When the reflectance of the object surface is lambertian, but the object is non-convex (e.g., part of the object surface is planar, see Fig.3), the *orientation-consistency* cue alone is not enough for reconstruction with B-UPS. In Fig.3 (a), two parts of the surface are planar. Within each planar surface, if two individual surface points have equal albedo, the *orientation-consistency* will always fail in reconstruction under the distant lighting. However, an alternative , called *local-orientation-consistency* cue, can be used to overcome this problem. This cue states that, under the non-distant lighting conditions, the local areas of the planar parts of the object surface are orientation-consistent, see Fig.3 (b). The *local-orientation-consistency* cue indicates that when the planar surface is illuminated by non-distant lighting, the irradiance of its surface points are different even for uniform albedo case. In Fig.3 (b), the light can also be regarded as point and non-point light source. Similar to Case 1, it is easy to check that the LOC is valid for both types of light sources.



(a)(b)

Fig. 3. The illustration of Case 1. In (a), the light source is regarded as point light source whereas, in (b), it is non-point light source.

Case 3: Non-Lambertian Reflectance and Convex Object. Torrance-Sparrow model is a physically-based reflectance model that consists of both the Lambertian part and, more importantly, a specular lobe of varying sharpness and strength. This model was based on a physical model of the roughness of surfaces and can capture the reflectance properties of a wide variety of non-Lambertian surfaces. It is assumed that the non-Lambertian reflectance in this paper is Torrance-Sparrow model. The pixel intensity derived from the Torrance-Sparrow model when a single distant light source illuminates the object is given by

$$I_{TS} = \alpha_d \|\mathbf{s}\| \cos\theta_i + \alpha_s \|\mathbf{s}\| Q F(\theta', \eta) \frac{exp(-\nu^2 \theta_\alpha^2)}{\cos\theta_r} \tag{1}$$

where the description for these parameters in (1) is introduced in [7]. Fig.4 shows the geometry of the Torrance-Sparrow model.

Since the camera we use is a parallel short-baseline stereo rig, with the assumption that the camera is located far enough from the target object, it is reasonable that the viewing directions of the two cameras are approximately the same. That means that the pixel intensities of the same surface point from the two-view images are approximately the same. With this constraint, we have

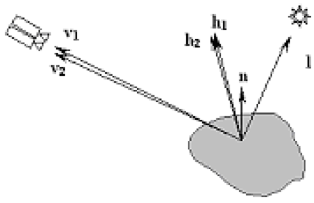


Fig. 4. The geometry of T-S model: $\mathbf{l}$ is the direction of light source. $\mathbf{v}_1$ and $\mathbf{v}_2$ are the viewing directions of the two cameras. $\mathbf{n}$ is the normal of the local surface patch. $\mathbf{h}_1$ is the bisector of $\mathbf{v}_1$ and $\mathbf{l}$, and $\mathbf{h}_2$ is the bisector of $\mathbf{v}_2$ and $\mathbf{l}$.

the following two observations similar to Case 1. (1) If the object is convex, the *orientation-consistency* alone is enough for reconstruction with B-UPS when the light is distant from the target object. (2) If the object is convex, the *orientation-consistency* alone is enough for reconstruction with B-UPS when the light is close to the target object. It is not hard to verify this observation by looking into Fig.2 (a) and (b) with non-Lambertian (T-S) reflectance model. By revisiting Fig.4 and Eq.(2), it is easy to see that the BRDF of the same point is almost the same when the camera is far enough. Therefore, the intensity of the pixels corresponding to the same surface point is approximately equal.

Case 4: Non-Lambertian Reflectance and Non-Convex Object. Similar to case 2, if the object is non-convex (e.g., with planar part), the *orientation-consistency* alone is not enough for reconstruction with B-UPS with distant lighting condition. However, the *local-orientation-consistency* cue can be incorporated into the B-UPS framework for complete reconstruction with close lighting condition.

3 Segmented Binocular Uncalibrated Photometric Stereo

A Segmented Binocular Uncalibrated Photometric Stereo (S-BUPS) algorithm is proposed in this section to improve the matching performance of B-UPS. Before the introduction of the S-BUPS, we give the motivation of this segmentation based B-UPS algorithm. When illuminated by distant light source, B-UPS can easily find the correspondence of the convex part of the object surface between the two views, but it fails in planar parts. However, the *local-orientation-consistency* cue with non-distant lighting can help B-UPS to find the correspondence for matching those points in planar areas. Therefore, our motivation is that whether we can find a method that can automatically detect those planar areas of the target object surface. Usually, it is a tough task to detect the planar areas of a scene if we have no prior information. As we know, the normals of all the planar-surface points are in the same direction. This naturally leads us to the uncalibrated photometric stereo because it is probably the most common method for normal estimation. Although currently we do not know the real-normals, however, we can easily compute the pseudo-normals by Singular

Table 1. $S - BUPS$ algorithm

Algorithm: $S - BUPS$

1. Input: two sets of stereo images: one is taken by stereo camera under distant lighting and the other is taken under non-distant lighting.

2. Based on the *orientation-consistency* cue, use both the two sets of stereo images for stereo matching

3. Using the distant-lighting stereo image set, calculate the pseudo-normal maps based on SVD.

4. Based on the pseudo-maps, calculate the variation of the pseudo-normals within a fixed-sized window area. Set a threshold for the variation to perform planar-area detection to find those planar areas. Label the segmented areas corresponding to those piece-wise planar surfaces.

5. Based on the *local-orientation-consistency* cue, use only non-distant lighting stereo images for stereo matching for those marked points which are detected as planar surfaces. For those areas which are detected as non-planar areas, keep the matching result of step **2** unchanged

6. Output: Disparity (depth) images.

Value Decomposition (SVD). Because the real-normals can be obtained from the pseudo-normals up to the same invertible transformation, those surface points with equal real-normals must have equal pseudo-normals in the pseudo-normal map. As long as we can find an small area within which each point has equal pseudo-normal, this area can be regarded as a small planar surface, see the detailed S-BUPS algorithm in Table 1.

4 Experiment Results

In our experiment, two objects are used as the target objects for our B-UPS and S-BUPS. They are a plaster Angel (lambertian non-convex) and a porcelain Fish (non-lambertian and non-convex). For each of them, we have captured 20 pairs of distant-lighting images and 15 pairs of non-distant-lighting images. Fig.5 shows two pairs of images for each object.

4.1 Pseudo-normal Segmentation for Planar-Area Detection

We use SVD decomposition to compute the rank-3 pseudo-normal images for both the left- and right-view images for the two objects. The first two rows of Fig. 6 show the obtained pseudo-normal maps which correspond to x, y, and z components for the two objects. According to the notion presented in Section 3 that the true normal can be obtained from the pseudo-normal up to the same transformation, we choose those small areas (a window with a size of 3×3 or 5×5) whose variation in pseudo-normal is smaller than a threshold as planar areas. The last four rows of Fig. 6 show the planar-area detection results for the scenes of Fish and Angel, where columns (a) to (c) are the results for the Fish

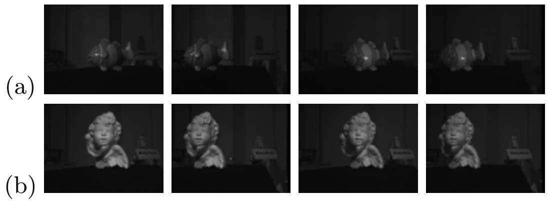


Fig. 5. Fish and Angel stereo image sets: (a) Fish (non-lambertian and non-convex), (b) Angel (lambertian non-convex)

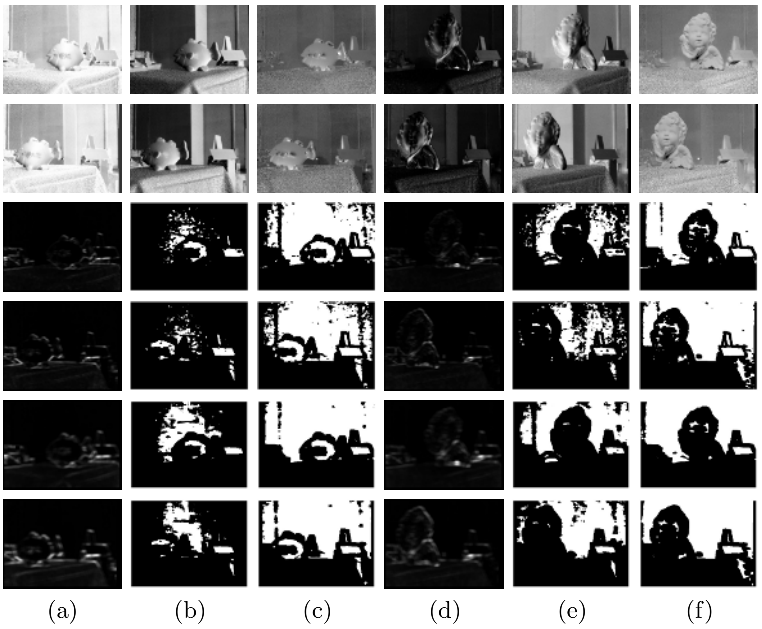


Fig. 6. Fish and Angel pseudo-normal images for the two views and planar-area detection results

scene and columns (d) to (e) correspond to the Angel scene. The two middle images in (a) show the variation maps for the left- and right-view when the window size is 3×3. The two bottom images in (a) show the variation maps for the left- and right-view when the window size is 5×5. The two middle images in (b) show the detected planar-areas with the threshold of 0.0001 in the variation of pseudo-normal for the two views (3×3) and the two bottom images in (b) show the detected planar-areas with the threshold of 0.0001 in the variation of pseudo-normal for the two views (5×5). The two middle images in (c) are the detected planar-areas with the threshold of 0.0003 for the two views (3×3) and the two bottom images in (c) are the detected planar-areas with the threshold of 0.0003 (5×5). The last four rows of Fig.6 (d) to (f) correspond to the detected results for

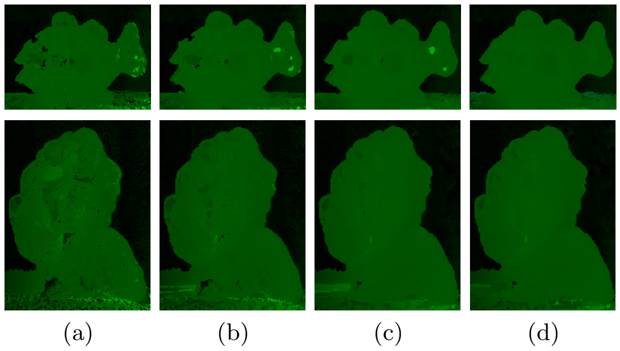


Fig. 7. Fish and Angel disparity images

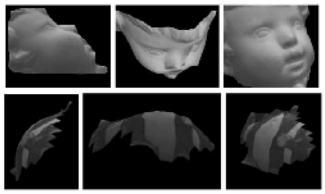


Fig. 8. Reconstructed 3D models for Fish and Angel

the Angel scene with similar settings, i.e., the two middle rows corresponding to the 3×3 window while the two bottom rows corresponding to the 5×5 window.

4.2 Stereo Matching with *OC* in S-BUPS

With *OC* and *LOC* cues, we show the stereo matching result for the two object using images taken under distant lighting. The metrics used in stereo matching are the Sum-of-Absolute-Differences (SAD) and window-based SAD (wSAD). The SAD is defined as the sum of the absolute value of the difference of $[p_L^1, p_L^2, \cdots, p_L^M]$ and $[p_R^1, p_R^2, \cdots, p_R^M]$ at the location i of left image sequence and the location j of the right-camera sequence, i.e.,

$$Mc(i,j) = sum(abs([p_L^1, p_L^2, \cdots, p_L^M]^T - [p_R^1, p_R^2, \cdots, p_R^M]^T))|_{i,j} \tag{2}$$

wSAD is to compute the sum of the absolute value of the differences of the (i,j) pixel-pair and its neighboring pixel-pairs. In our experiment, the window size is 3x3 or 5x5 respectively for each object. The stereo matching results are shown in Fig.7, where column (a) corresponding to the disparity images using SAD metric and *OC* constraint, (b) corresponding to the results using 3x3 window-based SAD and *OC* cue, (c) corresponding to the results using 5x5 wSAD and *OC* cue, (d) corresponding to the results using 5x5 wSAD, *OC* and *LOC* cues. From Fig.7, wSAD is generally better than SAD metric in computing the disparities. The results with *OC* and *LOC* constraints are better than the ones with only *OC* cue. Fig. 8 shows the reconstructed 3D Fish and Angel.

5 Conclusions

A framework of B-UPS is given for accurate stereo matching for lambertian and non-lambertian objects. By a planar-area detection based on a pseudo-normal-map segmentation scheme, the OC for distant-lighting condition and LOC for non-distant lighting condition are used together for stereo reconstruction.

References

1. Hertzmann, A., Seitz, S.: Example-based photometric stereo: Shape reconstruction with general, varying brdfs. IEEE Trans. on PAMI **27** (2005) 1254–1264
2. Hayakawa, H.: Photometric stereo under a light source with arbitrary motion. Journal of Optical Society of America, A (1994)
3. Yuille, A., Snow, D., Epstein, R., Belhumeur, P.: Determining generative models of objects under varying illumination: Shape and albedo from multiple images using svd and integrability. IJCV **35** (1999) 203–222
4. Belhumeur, P., Kriegman, D., Yuille, A.: The bas-relief ambiguity. IJCV **35** (1999) 33–44
5. Drbohlav, O., Sara, R.: Specularities reduce ambiguity of uncalibrated photometric stereo. In: ECCV. Volume 3. (2002) 46–60
6. Drbohlav, O., Chantler, M.: Can two specular pixels calibrate photometric stereo? In: ICCV. (2005) 568–574
7. Georghiades, A.: Incorporating the torrance and sparrow model of reflectance in uncalibrated photometric stereo. In: ICCV. Volume 2. (2003) 591–597

Empirical Evaluation of a Visual Interface for Exploring Message Boards

Beomjin Kim, Philip Johnson, and Jason Baker

Department of Computer Science
Indiana University-Purdue University
Fort Wayne, IN, U.S.A.
kimb@ipfw.edu

Abstract. This paper introduces a method that presents a number of characteristics of threads in a discussion forum through graphical illustrations. This technique brings together visual components, such as dimension, color, intensity, and position to present multiple aspects of a thread including the amount of information, popularity, activities, comparative value, and tenure of the thread. This high visual abstraction of threads allows us to display a large number of threads showing overall properties of the contents on a limited screen space. These proposed visualization techniques will assist the user to filtering noisy threads effectively from threads having important features. We have conducted an experimental study, which compares the effectiveness of the developed visual interface to a traditional text-based interface. The experimental study has shown that the user's search speed and accuracy in finding noticeable threads from a huge collection of threads has improved significantly by using the visual navigation tool.

1 Introduction

The internet has positioned itself as an indispensable tool for accessing massive collections of digitized information. This important communication media allows people to construct many forms of online social spaces such as Usenet newsgroups, message boards, blogs, and virtual environments. Traditional bulletin boards, which organize large amounts of text-based information in the form of threaded messages, have been actively used for social interaction and information sharing on a wide range of subjects. This valuable resource provides us with useful information, but this social space is commonly overloaded with contributions and innocuous information which impedes users' ability to find this information.

Current text-based interfaces present a list of threads using primitive information such as title, number of messages, references, thread initiator, and last posting date. Only a small fraction of this list is displayed. This is neither effective when accessing a large compilation of resources, nor can the users evaluate the quality of contents without reading portions of the threads. In order to address such limitations, researchers have applied visualization techniques that show visual abstractions of features of threads [1, 2, 3]. Studies have shown that users' reviewing speeds and insights into information are improved by exploiting their cognitive activities when exploring large information spaces [4, 5, 6, 7].

Several types of information filtering techniques have been explored to eliminate noisy threads from valuable threads by means of analyzing content or social interactions within these social spaces. Content-based filters suggest major discussion themes or semantic structures of an archive of articles after analyzing its content [10, 11]. Social filtering analyzes the behavioral aspects of authors over time to estimate influential sources of information by investigating the historical participations in social interactions [1, 3]. Perry and Donath proposed an anthropomorphic visualization that illustrates historical information about the population of an online social space [17]. In this system, human shape icons symbolize activities and roles of participants, emotional tone of messages, and the connections to other icons represent the social networks. A study analyzing the behavior of authors reported that the quality of a message is highly correlated with the frequency, longevity, and amount of information its author has contributed to the Usenet Newsgroup [12]. Collaborative filtering, which is a type of content-based filtering, utilizes the opinions of other readers to predict the value of content [13]. The accuracy of this assumption is highly correlated with the number of evaluators and their preferences. Note that there are many more passive readers than active writers in online discussion forums. The estimated value of threads will be unreliable when there are not a large enough number of evaluators [14].

Presenting a large volume of data on limited screen space is a challenging task without the assistance of visual abstraction. Previous studies have proposed visualization techniques to present features of message boards compactly and informatively [1, 2, 8]. The PeopleGarden project used a garden metaphor to visualize overall activities of users on a bulletin board [2]. In this activity portrait, individual authors are mapped to flowers and other attributes such as petals, color, and flower height represent the author's profile (i.e. the level of activity, author's tenure, and posting history). ForumReader has combined data visualization with automatic topic extraction techniques to provide visual overviews of conversations [15, 16]. This method lists thumbnails symbolizing messages with indentation along the horizontal axis that represents structure of the conversation. The thumbnail color indicates the quality of the content posted by conversation participants. The system executes contents searching by allowing the user to supply queries and then highlights all related messages in the forum. A recent paper introduces a technique for visualizing threads of message boards on mobile devices [8]. By using a treemap[9], this study renders threads in bulletin boards as colored rectangles on a reduced screen size. The size of a rectangle is proportional to the number of articles in a thread and the color represents the activity of the thread or relevancy of a thread to a given query. Their pilot study reported that applying visual interfaces to text-based message boards looks highly promising in assisting users' search activities.

These studies have suggested ways of presenting various aspects of discussion forums and their experimental results support the efficiency of exploiting visual interfaces to access larges amount of disparate information. Meanwhile, current approaches have focused on delivering only certain features associated with message boards. There are still many other valuable attributes for recognizing meaningful contents from noise. The main goal of this research is the development of a visual interface that helps the user to recognize worthwhile messages from a large collection of threads. This paper introduces visualization techniques that transform multiple aspects of threads to a graphical illustration while addressing the drawbacks in existing systems. The developed system called Discussion Forum Visualizer (DifVis), achieves higher visual abstraction by employing multiple visual components and presenting those relations intuitively. This visual overview of message boards will assist the user in evaluating the value of threads and correlations of threads while decreasing access to the underlying content of threads. We have conducted experiments to evaluate the effectiveness of DifVis as a visual interface for discussion forums.

The remainder of this paper is structured as follows. Section 2 introduces concepts and algorithms for transforming multiple attributes of threads to a visual representation. Section 3 explains supporting tools equipped in DifVis. Section 4 discusses the experiments and performance results of the proposed method. In Section 5, we discuss issues identified via experiments and suggest future works to be studied further. Finally, the last section presents the conclusions of this work.

2 Visualizing Procedures

The proposed system executes two major processes to transform major features of threads into visual notations. Each thread in the forum is mapped to a square shape object whose dimension is defined relative to the magnitude of the associated thread and the temporal location of the thread in a message board. This method exploits color-coding to represent the popularity, activities, and tenure of a thread where the color intensity represents the significance of each component. The following subsections describe in detail the mechanism of visual transformations.

2.1 Visualizing Magnitude of Threads

As suggested in [12], the number of postings to a thread shows a close correlation to the quality of that thread. To display this significance graphical, the DifVis system maps the total number of words in the forum ($\sum_{j=0}^{n} T_j^W$) to the total number of pixels available to the viewport (V_{size}). The magnitude of thread i is proportional to the magnitude of the entire forum and is relative to V_{size}. This transformation can be expressed as

$$T_i^M = \sqrt{(T_i^W \cdot V_{size}) / \sum_{j=0}^{n} T_j^W} \; . \tag{1}$$

where T_i^M is the magnitude of thread i, and T_i^W is the total number of words in thread i. This magnitude defines the size of a square object representing a thread. The system then considers the relationship between magnitude and time. Threads that have a substantial lifetime will overpower threads that have not had the opportunity to build up as much information. To lessen the effects of this the DifVis system then applies a perspective projection where the temporal domain is mapped to the Z-axis. We define the position of T_i along the Z-axis as the distance from the front of T_i to the center of projection (T_i^Z). This projection has the advantage in that it places higher significance on current information than on information from the past. In other words, threads at the beginning of the temporal domain will be displayed with its normal magnitude whereas threads that do not have recent activity will have its magnitude diminished. Eq.(2) shows this computation,

$$T_i^D = \alpha * T_i^M / T_i^Z . \tag{2}$$

where T_i^D is the dimension of thread i, and α is the distance from the center of projection to the projection plane. By moving the camera down the Z-axis the user can then control the time domain visible to them.

2.2 Visualizing References to Threads

The number of references to a thread is the number of readers who have viewed a thread's content. This is a useful form of collaborative filtering which can be used to determine a given thread's quality by proxy instead of by content examination. Collaborative filtering can be a practical mechanism for distinguishing noise from constructive information in the information space [13]. It allows users to rapidly distinguish and remove noisy threads present in the discussion forum, by giving a lower value to threads with a low number of references.

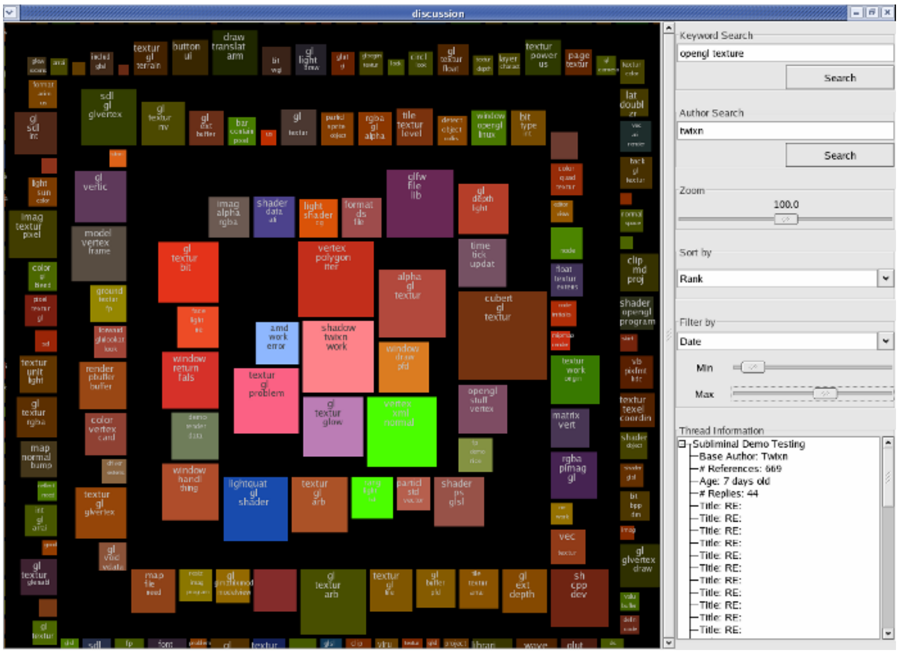
In the DifVis system, we represent the number of references with the green color component (Eq.(3)).

$$T_i^{G\in C} = \sqrt{(T_i^R / T_i^L) / (T_{\max}^R / T_{\max}^L)} * I_{\max} . \tag{3}$$

where $T_i^{G\in C}$ is the intensity of green component associated of thread i, T_i^R is the total number of references to thread i, T_i^L is the lifetime of the thread, $T^R{}_{max}$ and $T^L{}_{max}$ refer to the thread with the maximum number of references and it's associated lifetime, and I_{max} is the brightest intensity of green possible on the machine. To diminish the magnitude of the range, which can be distorted by special purpose threads, the square root was used to standardize the values [20]. Generally threads that have a long lifetime have a larger number of references. Dividing T_i^R by T_i^L ensures that the intensity distribution does not favor only threads with a long lifetime.

2.3 Visualizing Thread Tenure

The tenure of a thread allows users to estimate the integrity of a topic [12]. Subject matter which the population of users feel is insignificant will die quickly, whereas threads which can keep a population's interest will live longer. Defining the tenure as simply the amount of time between the postings of the first message and the last message can be deceptive. A thread which contains two messages posted at the ends of a given time period is not as interesting as a thread which has messages posted to it



Keyword Search
opengl texture
Search

Author Search
twixn
Search

Zoom
100.0

Sort by
Rank

Filter by
Date
Min
Max

Thread Information
Subliminal Demo Testing
Base Author: Twixn
References: 669
Age: 7 days old
Replies: 44
Title: RE:
Title: RE:
Title: RE:
Title: RE:
Title: RE:
Title: RE:
Title: RE:
Title: RE:
Title: RE:
Title: RE:
Title: RE:

Fig. 1. DifVis showing the visualized threads in a message board

every day over the same period of time. Therefore, to define a more meaningful representation of tenure, the DifVis system uses the number of days on which there was at least one message posted. The red color component is used to symbolize the tenure. The intensity of the red component is defined as follows

$$T_i^{R\in C} = (T_i^T * I_{max}) / T_{max}^T . \tag{4}$$

where $T_i^{R\in C}$ is the intensity of the red component of thread i, I_{max} is the maximum color intensity, $T_i{}^T$ is the tenure of thread i, and $T_{max}{}^T$ is the longest tenure of a thread in the discussion forum.

2.4 Visualizing Temporal Activities of Threads

Subjects of discussion and their popularity vary over time depending on the populations interests. Threads which summon a higher level of interest will be accessed by a larger percentage of the population and will therefore have more postings made to it. A thread that has a more common subject matter, or has a topic that is not entirely time dependant, will in general show a relatively constant posting activity over its lifetime. Meanwhile, the population's posting activities to a thread about a periodic issue will change considerable and abruptly. Especially the users' posting activity to a newly initiated thread regarding a current issue will show greater fluctuation.

In order to deliver the population's posting patterns, the DifVis system uses the blue color component. Turbulence is, in the DifVis system, the measure of variability of the number of postings per day to a thread, measured by the standard deviation (μ). Since users are generally more interested in the most recent topics, the DifVis systems weights μ based on when the fluctuations have occurred.

$$T_i^{Tu} = \sqrt{\sum_{a=0}^{n} \left((\overline{T}_i^{A} - T_i^{a})^2 * NORM(T_i^{aD}) \right) / \sum_{a=0}^{n} NORM(T_i^{aD})} \, . \tag{5}$$

where T_i^{Tu} is the turbulence of thread i, T_i^{a} is the number of postings on day a and $NORM(T_i^{aD})$ is the normalized length of time from today to a. The T_i^{Tu} of each thread linearly maps to the intensity of the blue color component where the brightest intensity is assigned to the thread that has the greatest amount of turbulence as

$$T_i^{B\in C} = (T_i^{Tu} * I_{\max}) / T_{\max}^{Tu} \, . \tag{6}$$

where $T_i^{B\in C}$ is the intensity of the blue component of thread i, I_{max} is the maximum color intensity, and T_{max}^{Tu} is the greatest amount of turbulence in the forum.

Figure 1 shows a visualization of a forum which contains approximately 200 threads over a 90 day time domain. The threads are displayed in a radial layout and sorted based on the average significance of the four visual components.

3 Supporting Tools

The DifVis system provides interactive tools, keyword search, attribute filtering and sorting, and Focus+Context to increase the effectiveness of the system. Keyword searching can be utilized by users to highlight threads containing information requested. Filtering removes noisy threads based upon user-defined thresholds for the attributes. Sorting allows the user to visually distinguish noise from value. Focus+Context enables users to drill down to more detail such as associated keywords. Thread keywords are defined using content analysis, suffix-stemming and stop-word removal, where the keyword's size is scaled proportional to its frequency in the thread.

4 Evaluation

4.1 Experimental Environments

The experiment was conducted in an isolated PC-Lab where each computer was equipped with a 2.8GHz Pentium 4 Intel CPU and 1.0 GB main memory. The time required for analyzing and retrieving a huge amount of threads can be a variance in this experiment. For a more accurate comparison between the DifVis and traditional systems, properties of threads' are pre-analyzed and the metadata is stored. This allows both applications to generate output with minimal lead-in time. Twenty-two experimental participants have been recruited from the authors' university. They had no difficulties using the computers, no problems in color perception, and the majority of the participants have experience using some type of online social environment. Three distinct data sets were used covering mutually exclusive topics from separate message boards and having variations in the attributes' distributional qualities.

4.2 Experimental Procedures

Using two randomly selected experimental data sets, each participant classified the significance of the threads using the DifVis system for one and the traditional text-based system (TextB) for the other. Although the DifVis provides various visual navigation tools, for an accurate evaluation of the proposed approach these tools were disabled. Based on their judgments, participants were asked to identify as many threads as possible that possess significant characteristics within a five-minute period. For each application when the user clicks on a thread the rank of each visual attribute defined above is displayed.

Although both systems were intuitive, an orientation of both systems was given to the participants before starting the experiment. This introduction explained the basic functionalities that are provided by both systems along with the significance of the visual abstractions. Participants were given enough time to be familiar with both systems and fully comprehend their assigned tasks.

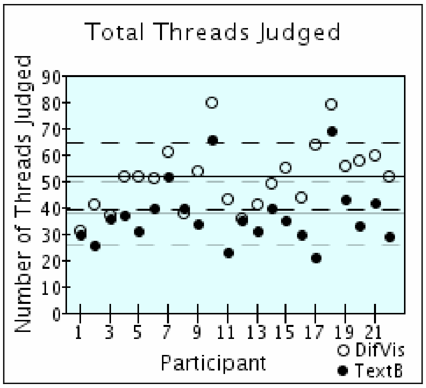
4.3 Experimental Results

The table below presents experimental results comparing participants' speed and accuracy in categorizing the attributes utilizing the two systems. The speed of identifying significant threads was evaluated by measuring the total number of categorized threads within the experimental time period listed in the second column of table 1. The accuracy of identifying threads having outstanding attributes was measured in the following manner. Let us assume a participant marked an attribute of a thread as significant. When the actual percentile value of that attribute is ranked within the top λ% of the data set, this judgment is counted as a correct identification. Otherwise, the thread is considered incorrectly categorized. The second and third column of the table below show the accuracy of all judged attributes with two different λ's 5 and 10.

Table 1. Comparative experimental results showing the swiftness and accuracy of recognizing noticable threads between the DifVis and TextB interfaces

	Judging speed (Mean ± SD)	Accuracy (Within top 5%)	Accuracy (Within top 10%)
DifVis	51.6 ± 12.7 threads	70.7 %	84.3 %
TextB	37.4 ± 12.0 threads	35.4 %	57.1 %

Figure 2.a shows the results of judging speed where the X-axis represents the individual participant and the Y-axis denotes the total number of categorized threads. The solid line stands for the mean and dashed lines represent the standard deviation. Figure 2.b shows the accuracy of all judged characteristics and accuracy for individual attributes. The X-axis shows the four attributes associated with each thread and the Y-axis is the number of attributes that are categorized correctly and incorrectly.



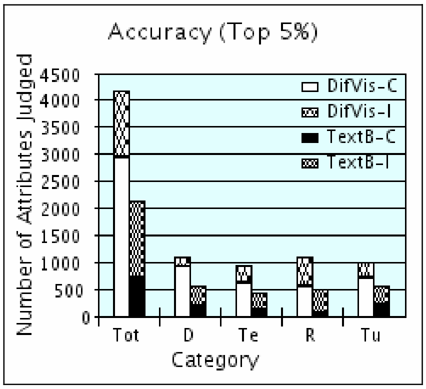


Fig. 2. **(a)** The judging speed of individual participant**.** **(b)** Accuracy of categorizing four attributes of judged threads where Tot is the total number of attributes judged, D is Dimension, Te is Tenure, R is References, Tu is Turbulence, -C is correct, and -I is incorrectly judged.

The experimental outcomes from these testing data sets were as expected. Participants who used the DifVis system categorized approximately 1.38 times as many threads as those who used the TextB system. The accuracy of identifying important threads ranked in the top 5% improved significantly, about 2 times when they used the visual interface.

5 Discussion and Future Works

We conducted a post experiment survey to collect the participants' opinions regarding the performance of the developed system as a visual interface. The questionnaire asked the participants' views concerning the DifVis system in comparison with the TextB system for distinguishing threads with remarkable attributes. The users' feedback about the DifVis as a visual navigation tool was positive. Table 2 shows the survey results for selected questions.

The DifVis system uses three primary color red, green, and blue to encode the value of attributes. Each color component varies within a predefined intensity range. The sensitivity of the eye to the same intensity of primary colors is different. The Spectral-response function suggests that the human eye is most sensitive to greenish-yellow light [18]. In order to reflect the importance of collaborative filtering suggested in previous studies [13, 16], the Difvis system employed green to represent the degree of references to a thread. As shown in Fig. 2, experimental participants judged more threads where they felt the reference was a significant attribute than threads where they felt the turbulence was significant attributes, but the actual significance was overestimated more often for references. It is a challenging task to ensure that attributes with unequal significance are assigned to colors of appropriate spectral intensity. In the future, further research is expected to use color-coding while considering the comparative importance among attributes.

Table 2. Summarized results from post experiment survey

Items	Means (Out of 5)
Overall satisfaction using DifVis over traditional interface	4.2
Improved ability to recognize the major subjects of discussions	4.3
Helpfulness to predict the value of threads	4.4
Easiness of finding threads having larger information	4.4
Easiness of finding threads having active users' participation	4.7
Ease of finding threads having longer lifetime with recent contents	4.5
Ease of finding threads having fluctuations in users' participation	4.3
Preference of using visual interface in the future	4.3

One of the major difficulties in executing a systematic experiment is the lack of a gold standard to be compared to for the evaluation of the system. The users' judgment about the importance of attributes was compared to the relative order of values of attributes. The rank of the attribute's value can not properly represent the significance of threads. A high value for an attribute in one forum may be a low value for that same attribute in another forum. Although we executed experiments with apparent answers, the performance of DifVis can be assessed more objectively when we conduct experimental studies with a standardized data set similar to test collections provided by National Institute of Standards and Technology [19].

6 Conclusions

This paper introduces a visual interface for presenting multiple aspects of threads in message boards through a graphical illustration. By integrating color-coding and other visual components to encode major features of threads, our system achieves higher visual abstraction while enhancing deficiencies of existing methods. The visual representation of threads uses the limited screen space efficiently. It helps the user to predict the quality of information in the thread and to understand the correlations among threads. Eventually it will expand the user's ability to find desired information from a workspace both vast and noisy. The experimental study has supported the feasibility of using this proposed visualization technique for exploring message boards. The post experiment survey shows that the visual navigation tool, DifVis, is well-liked by users. This study also suggests the future task that will be investigated further to make the DifVis system a practical application.

References

1. Donath, J.: A semantic approach to visualizing online conversations. Communication of the ACM, Vol. 45, No. 4 (2002) 45–49
2. Xiong, R., Donath, J.: PeopleGarden: Creating data portraits for users. Proceedings of the Twelvesth Annual ACM Symposium on User Interface Software and Technology, (1999) 37–44

3. Boyd, D., Lee, H-Y., Ramage, D., et al: Developing legible visualizations for online social spaces. Proceedings of the Hawaii International Conference on System Sciences, (2002) 115–124
4. Card, S., Mackinlay, J., Shneiderman, B.: Readings in Information Visualization: Using Vision to Think. Morgan Kaufmann (1999)
5. Veerasamy, A., Heikes, R.: Effectiveness of a graphical display of retrieval results, Proceedings of the Twentieth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (1997) 236–245
6. Kim, B., Johnson, P., Huarng, A.: Colored-sketch of Text Information. Journal of Informing Science, Vol. 5, No. 4 (2002) 163–173
7. Donath, J., Vieas, F.: The chat circles series: explorations in designing abstract graphical communication interfaces. Proceedings of the Conference on Designing Interactive Systems: processes, practices, methods, and techniques, (2002) 359–369
8. Engdahl, B., Köksal, M., Marsden, G.: Using treemaps to visualize threaded discussion forums on PDAs. Proceedings of the SIGCHI Extended Abstracts on Human Factors in Computing Systems, (2005) 1355–1358
9. Johnson, B., Shneiderman, B.: Tree-Maps: a space-filling approach to the visualization of hierarchical information structures. Proceedings of IEEE Visualization, (1991) 275–282
10. Sack, W.: Conversation map: a content-based Usenet newsgroup browser. Proceedings of the Fifth International Conference on Intelligent User Interfaces, (2000) 233–240
11. Newman, P.: Exploring discussion lists: steps and directions. Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital libraries, (2002) 126–134
12. Fiore, A., Tiernan, S., Smith, M.: Observed behavior and perceived value of authors in usenet newsgroups: bridging the gap. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, (2002) 323–330
13. Konstan, J., Miller, B., Maltz, D. et. al: GroupLens: applying collaborative filtering to Usenet news. Communications of the ACM, Vol. 40, No. 3, (1997) 77-87
14. Allen, R.: User models: Theory, method, and practice. International Journal of Man-Machine Studies, Vol. 32, (1990) 511–543
15. Wattenberg, M., Millen, D.: Conversation thumbnails for large-scale discussions. Proceedings of the SIGCHI Extended Abstracts on Human Factors in Computing Systems, (2003) 742–743
16. Dave, K., Wattenberg, M., Muller, M.: Flash forums and forumReader: navigating a new kind of large-scale online discussion. Proceedings of the ACM Conference on Computer Supported Cooperative Work, (2004) 232–241
17. Perry, E., Donath, J.: Anthropomorphic Visualization: A New Approach for Depicting Participants in Online Spaces. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, (2004) 1115–1118
18. Foley, J., van Dam, A., Feiner, S., et al: Computer Graphics: Principles and Practice. 2nd edn. Addison-Wesley Publishing Company, (1996)
19. The Text REtrieval Conference Web site. http://trec.nist.gov/
20. Hand, D., Mannila, H., Smyth, P.: Principles of Data Mining. MIT Press, (2001)

Direct Estimation of the Stereo Geometry from Monocular Normal Flows

Ding Yuan and Ronald Chung

Department of Automation & Computer-Aided Engineering
The Chinese University of Hong Kong
Shatin, Hong Kong, China
{dyuan, rchung}@acae.cuhk.edu.hk

Abstract. The increasing use of active vision systems makes it necessary to determine the relative geometry between the cameras in the system at arbitrary time. There has been some work on on-line estimation of the relative camera geometry parameters. However, many of them are based on epipolar geometry, motion correspondences, or even presence of some calibration reference objects in the scene. In this paper, we describe a method that allows the relative geometry of two cameras be estimated without assuming that their visual fields picture the same object, nor that motion correspondences in each camera are fully estimated beforehand. The method starts from monocular normal flows in the two cameras and estimates the relative geometry parameters without evening accessing the full optical flows. Experimental results are shown to illustrate the performance of the method.

1 Introduction

Active camera systems, often called "head-eye" systems, typically consist of a pair of cameras whose motions can be individually controlled. The active camera system platform could be more effective in achieving 3D reconstruction when stereo vision and dynamic vision are integrated. However, the relative geometry between the cameras could be varying should each camera have its own degree of freedom, and needs be estimated in an on-line manner. There have been much work on online estimation of the intrinsic parameters (the focal length and principal point) [[2] [12] [13]], which are parameters related to the inherent settings of the cameras. In this work we only focus on the on-line estimation of the camera-to-camera geometry, and make the assumption that the intrinsic parameters remain unchanged or are available by the application of the above methods.

There has been some work on estimating the binocular geometry in on-line manner. However, many of them require certain specific objects appearing in the scene. There are other techniques [4] [8] [13] that do not need the presence of calibration reference object, but they require the accessibility of either cross-camera feature correspondences [1] [9] [11] [13] or motion correspondences [3]. While cross-camera correspondences require that the cameras have much in common in what they picture, estimating motion correspondences (i.e., full optical flow) from the directly observable normal flow is an ill-posed task due to the aperture problem and requires the introduction of heuristics like smoothness which could lead to errors.

We see the binocular geometry estimation problem in the following light. If binocular geometry could be deduced from full optical flows, and full optical flows from normal, there likely exists a mechanism that allows the binocular geometry be estimated from the normal flows directly. In this paper, we propose a method to estimate the binocular geometry from the monocular normal optical flows by borrowing the concept of motion field introduced in [5][6]. Our work aims at arriving at a solution of recovering the relative geometry of two camera heads without assuming the presence of calibration objects or specific shapes or features in the imaged scene, allowing the possibility of on-line application. Furthermore, no restriction is to be imposed on the viewing directions of the respective cameras, allowing the relative geometry be estimated despite possibly no overlap in the visual fields of the respective cameras.

This paper is structured as follows. Section 2 gives a brief review of the field models we need for the image domain. In section 3 we describe our method on binocular geometry estimation. Experimental results will be shown in section 4. Finally in section 5, we give a conclusion and put forward the future work.

2 Vector Fields Induced by Arbitrary Axis for Image Domain

Here we review briefly the motion field models for the image domain, which were introduced by Fermüller and Aloimonos [5] [6].

2.1 Copoint and Coaxis Vector Field

Suppose the normalized planar image surface is positioned perpendicular to the optical axis. We choose an axis **s**= [A B C] passing through the image plane with the intersection point S=[A/C B/C]. The conic sections on the image plane are generated by a family of cones with respect to axis **s**. The s-copoint vector field is defined as the field with vectors perpendicular to the tangent of conic sections at each image point:

$$[M_x, M_y]=[(-A(y^2+f^2)+Bxy+Cxf),(Axy-B(x^2+f^2)+Cyf)] \tag{2.1}$$

where [M_x, M_y] is the vector assigned at image point[x, y] in s-copoint vector field for a given axis **s**= [A B C], and f is the focal length of the camera, as shown in Fig.1 (a). If the camera rotates about the direction of s-axis, then the full optical flows in the image domain due to this rotation will be just orthogonal to the s-copoint vector field.

Similarly, we take an arbitrary axis **p**=[A B C] passing the image plane at the point P=[A/C B/C], and then draw a series of lines starting from point P. The p-coaxis vector field is defined as the field with vectors perpendicular to the lines passing through point P, as shown in Fig. 1 (b).

$$[M_x \quad M_y]\frac{[-y+B/C \quad x-A/C]}{\sqrt{(x-A/C)^2+(-y+B/C)^2}} \tag{2.2}$$

where $[M_x, M_y]$ is the vector assigned at image point$[x, y]$ in p-coaxis vector field for a given axis $\mathbf{p}=[A\ B\ C]$. If the camera translates in the direction of p-axis, full optical flows in the image domain will then be orthogonal to the p-coaxis vector field.

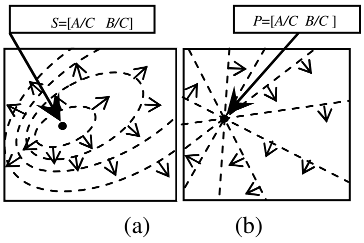


Fig. 1. (a) s-copoint vector field (b) p-coaxis vector field

Actually, s-copoint vector field and p-coaxis vector field are equivalent and can be generated by an arbitrarily given axis. In the following sections we will only discuss the camera motion under the s-copoint vector field for an example.

2.2 Field Models for Camera Motion Estimation

Here we review how the above vector fields induced by arbitrary s-axes could aid in recovering the camera motion from video data. The idea is adopted from what is presented in [5][6].

Suppose full optical flows are computed at each image point when the camera undergoes a pure translation. For a given s-axis, an s-copoint vector field is generated by assigning a specific vector to each image point according to equation (2.1). Then we examine the dot product of the full optical flow and the vector defined by the s-copoint vector field at each image point, and label the image point a '+' if its dot product is positive and vice versa. Finally a pattern with the'+' and '-' labels in the image domain is generated. According to the '+' and '-' labels, the image plane will be divided into two regions (Fig.2 (a)): one is called positive region with all the dot products positive; and the other is the negative region with all negative dot products. Also in this pattern there exists a second order curve, called zero-boundary, between the positive and negative regions, on which the image points have optical flows just orthogonal to the vectors in the vector field. Moreover, this second order curve is a function of the focus of expansion (FOE), which exactly describes the translational direction of the camera.

Fig.2 (b) illustrates the positive-negative pattern generated in the same way described above, when the camera takes only pure rotation. Let's describe the rotation using a vector $\boldsymbol{\omega}=[\alpha\ \beta\ \gamma]$, where the direction of $\boldsymbol{\omega}$ is the axis that the camera rotates about, and the length of $\boldsymbol{\omega}$ is the rotational angle. Different from the pattern caused by camera's pure translation, the zero-boundary on the pattern (Fig.2 (b)) is now a straight line which is only determined by the ratios α/γ and β/γ.

If the motion of the camera includes both translation and rotation, according to Fermüller and Aloimonos' theory, the positive-negative pattern can be simply obtained by the addition of the pure translational pattern (Fig.2 (a)) and the pure rotational pattern (Fig.2 (b)), if following the rule: Positive+Positive=Positive; Negative+Negative =Negative; Positve+ Negative =Don't know (depends on the structure of the scene), as shown in Fig. 2 (c).

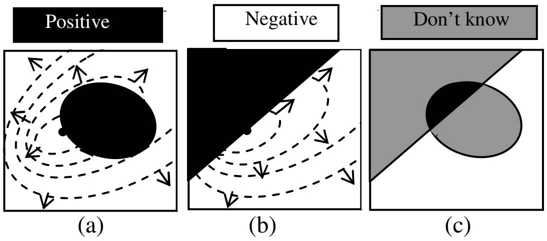


Fig. 2. s-copoint positive-negative patterns. (a) Camera undergoes pure translations. (b) Camera undergoes pure rotations. (c) Camera takes an arbitrary motion.

However, only normal flows can be computed directly from the image sequences. Fortunately, the positive-negative pattern still can be generated by only using the normal flows in specific directions for a given s-axis according to Fermüller and Aloimonos' theory [5] [6].

Since the zero-boundaries in the positive-negative patterns carry the information of the camera motion, it is possible to estimate the binocular geometry by locating these zero-boundaries when the stereo-rig as a whole undergoes a motion, without recovering the explicit ego-motion of each camera. Therefore, estimating the binocular geometry can be realized by only analyzing monocular normal flows.

3 Stereo Geometry Estimation

In this section, we will present the novel method on estimating binocular geometry using monocular normal flows.

In Fig. 3, we suppose the stereo rig moves as a whole to let stereo geometry ($\mathbf{R}_x$, $\mathbf{t}_x$,) remain the same during the motion. ($\mathbf{R}_A$, $\mathbf{t}_A$,) and ($\mathbf{R}_B$,$\mathbf{t}_B$) represent the motions of camera A and camera B respectively.

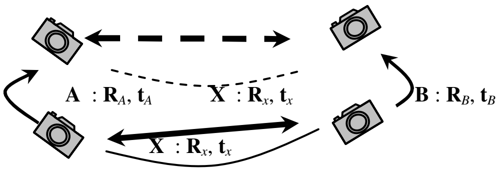

A : $\mathbf{R}_A$, $\mathbf{t}_A$
X : $\mathbf{R}_x$, $\mathbf{t}_x$
B : $\mathbf{R}_B$, $\mathbf{t}_B$
X : $\mathbf{R}_x$, $\mathbf{t}_x$

Fig. 3. Stereo geometry estimation of active binocular head

We use a 4×4 matrix **X** to represent the stereo geometry. Suppose the motions of cameras are **A** and **B**, then we have **AX=XB** which could be broken into:

$$\mathbf{R}_A\mathbf{R}_x = \mathbf{R}_x\mathbf{R}_B \text{ (or } \boldsymbol{\omega}_A = \mathbf{R}_x\boldsymbol{\omega}_B \text{ in vector form)} \tag{3.1}$$

$$(\mathbf{R}_A - \mathbf{I})\mathbf{t}_x = \mathbf{R}_x\mathbf{t}_B - \mathbf{t}_A \tag{3.2}$$

where $\mathbf{R}_x$, $\mathbf{R}_A$, $\mathbf{R}_B$ are 3×3 rotational components in **X**, **A**, **B**, and $\mathbf{t}_x$, $\mathbf{t}_A$, $\mathbf{t}_B$ are the translational vectors, $\boldsymbol{\omega}_A$ and $\boldsymbol{\omega}_B$ are the rotational components of camera A and B respectively in the vector form.

Obviously the stereo geometry can be calculated by equation (3.1) and (3.2) if all $\boldsymbol{\omega}_A$, $\boldsymbol{\omega}_B$ and $\mathbf{t}_A$, $\mathbf{t}_B$ are available. However, in our novel algorithm only partial information of motions achieved by locating zero-boundaries on the positive-negative patterns (Fig.2) is sufficient to calculate stereo geometry. Even so, locating zero-boundaries precisely is itself a great challenge. Firstly of all, for an arbitrarily given s-axis, only a few image points, where their normal flows are exactly along the direction of the s-copoint vector field, would be valid candidates and taken into account to generate the positive-negative pattern. Therefore, the positive-negative pattern actually has sparse positive and negative candidates, which causes the great difficulty in precisely locating the zero- boundaries. Furthermore, in Fig.2(c), there are two "Don't know" regions appearing in the pattern when camera takes an arbitrary motion. These regions result in more uncertainties in estimating zero-boundaries.

Aiming at the above two major problems, we propose our strategies. Firstly, we utilize more s-axes to make use of as many normal flows in the image as possible to improve the precision of the estimation. Secondly, some specific motions are applied to simplify the analysis on the patterns.

3.1 Estimation of $\mathbf{R}_x$

Estimation of $\mathbf{R}_x$ is the first step in calculating the stereo geometry. In this step, we let the stereo rig undergo the specific motion, pure translation, so as to reduce the complexity on locating the zero-boundaries in the positive-negative patterns.

When the stereo rig only has pure translations as a rigid body, motions of both cameras are pure translations. From (3.2) we have:

$$\tilde{\mathbf{t}}_A = \mathbf{R}_x\tilde{\mathbf{t}}_B \tag{3.3}$$

where $\tilde{\mathbf{t}}_A$ and $\tilde{\mathbf{t}}_B$ are both unit vectors representing the FOEs of the two cameras.

Equation (3.3) only provides two linear equations for $\mathbf{R}_x$. Hence, at least two translational motions in different directions possibly are the minimum to achieve a unique solution of $\mathbf{R}_x$ [10]. The rotational component of the stereo geometry is:

$$\mathbf{R}_x = \mathbf{U}_t \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(\mathbf{U}_t\mathbf{V}_t^{\mathrm{T}}) \end{bmatrix} \mathbf{V}_t^{\mathrm{T}} \tag{3.4}$$

where $\mathbf{K}_t = \mathbf{U}_t\ \mathbf{S}_t\ \mathbf{V}_t$ by SVD (singular value decomposition), and $\mathbf{K}_t = \sum_{\alpha=1}^{N} \tilde{\mathbf{t}}_A \tilde{\mathbf{t}}_B^{\mathrm{T}}$. N represents the number of pure translations in different directions. Equation (3.4) has a unique solution if and only if rank ($\mathbf{K}_t$)>1. We conclude that the unique solution of $\mathbf{R}_x$

always exits, if at least two pure translations of the stereo rig in different directions are available, that is, min (N) =2.

Now that the motion of the stereo rig only includes pure translation when capturing images, assume we adopt the s-copoint vector field to generate patterns from the normal flows calculated by the image sequences, and then both cameras will have the positive-negative patterns (like Fig. 2 (a)) containing only the positive and the negative regions divided by a second order curve, without including the "Don't know" regions. Obviously locating precise zero-boundaries on these positive-negative patterns is not a challenging task, as long as enough s-axes are used.

3.2 Estimation of $\mathbf{t}_x$ Up to Scale

Similar to the previous step of estimating the rational component $\mathbf{R}_x$, we apply another kind of specific motion to simplify the positive-negative pattern analysis. We choose pure rotations this time to the stereo rig to compute the baseline $\mathbf{t}_x$ of stereo geometry. However, only $\mathbf{t}_x$ up to scale is able to obtained, since we do not have any information on the 3D world beforehand.

Suppose the stereo rig has a pure rotation about an axis passing through the optical center of one camera, say optical center of camera A, then camera A only undergoes a pure rotation; while for camera B, it rotates about an axis passing through the optical center of camera A, and at the same time translates along the direction tangent to the baseline. In this case equation (3.2) is rewritten as:

$$(\mathbf{R}_A - \mathbf{I})\mathbf{t}_x = \mathbf{R}_x \mathbf{t}_B \tag{3.5}$$

where $\mathbf{R}_A$ is the rotation of camera A, vector $\mathbf{t}_B$ is the translation of camera B, and rank ($\mathbf{R}_A$- $\mathbf{I}$)=2, since it has a nonzero nullspace. We then rewrite (3.5) to a homogeneous system:

$$\widehat{\mathbf{A}}\tilde{\mathbf{t}}_x = \mathbf{0} \tag{3.6}$$

where $\widehat{\mathbf{A}}$, calculated by $\mathbf{R}_x$, $\mathbf{R}_A$ and $\tilde{\mathbf{t}}_B$, is a 2×3 matrix with rank($\widehat{\mathbf{A}}$)=1, and $\tilde{\mathbf{t}}_x$ is the normalized vector which represents the direction of the baseline. Therefore, at least two rotations are the minimum to compute a unique $\tilde{\mathbf{t}}_x$. We then apply SVD to the homogeneous linear system of equations (3.6) to estimate the optimal $\mathbf{t}_x$ up to scale.

Camera A, including only pure rotation, has the positive-negative pattern generated from its normal flows by a given s-axis with the characteristic as shown in Fig. 2 (b), where a straight line divides the image domain into positive and negative regions, without having the "Don't know" area. Thus, locating the straight line on such pattern is not a difficult task. We use vector $\boldsymbol{\omega}_A$=[α_A β_A γ_A] to represent the rotation of camera A. The ratios α_A/γ_A and β_A/γ_A will be obtained during locating this straight line (zero-boundary). The two ratios cannot describe full rotation $\boldsymbol{\omega}_A$, so the third component γ_A have to be estimated by using the method named "detranslation" presented in [6].

For camera B, the positive-negative pattern generated from normal flows is much more complex than the pattern of camera A, because of the two "Don't know" regions within the pattern (Fig.2(c)). However, the rotational component of camera B $\boldsymbol{\omega}_B$ can be computed directly from equation (3.1), since $\mathbf{R}_x$ is already calculated in the previous step and available now. Assume we apply s-copoint vector field, then the straight line of the two zero-boundaries on the positive-negative pattern is known to us if $\boldsymbol{\omega}_B$ is determined, consequently the other second order zero-boundary defined by FOE is easy to locate, even though there are two "Don't know" regions. The direction of the translational component $\mathbf{t}_B$ is determined during locating the second order boundary. $\mathbf{t}_x$ (up to scale) can be calculated by far from (3.6), since $\mathbf{R}_A$ and $\tilde{\mathbf{t}}_B$ are both computed from the positive-negative patterns.

4 Experimental Results

We have implemented the proposed method and tested it with both synthetic and real image data to investigate its performance. The whole algorithm includes two steps. Firstly, the stereo cameras undergo pure translations twice as a whole, and each time they moves in different directions. The rotational component $\mathbf{R}_x$ is computed in the first step. After that, we rotate the stereo rig twice according to two different axes passing through the optical center of one of the cameras to estimate $\mathbf{t}_x$ up to scale.

4.1 Experimental Results on Synthetic Data

The experiments on synthetic data aim to investigate the accuracy and precision of the algorithm. We firstly applied synthetic image data to evaluate the accuracy of the algorithm. Suppose the image resolution is 101×101.

Estimation of $\mathbf{R}_x$ Using Synthetic Image Data. After calculating the synthetic optical flows at each image point according to the given two translational velocities of the camera motions, we assigned each image point an arbitrary gradient direction to generate the normal flows for each camera. In the experiment on synthetic data, these normal flows are the inputs of our system to estimate the binocular geometry.

Given the first arbitrary s-axis, for instance **s**=[1 0 0], we got the first pattern as shown in Fig 4(a). Initially FOE is supposed possibly located in anywhere within the image frame. After investigating the pseudo FOEs 0.25 by 0.25 pixel, more than 1000 curves determined by these pseudo FOEs could divide the pattern into two regions well. Then we applied a second s-axis to examine whether these 1000 pseudo FOEs that had good performance in the first pattern would still perform well in the new pattern. We discarded wrong FOEs that had bad performance this time, and kept others to the next round of new s-axis, until all the possible FOEs located within a small area. The number of possible FOEs dramatically decreased when more s-axes are utilized. Then the center of all possible FOEs' was considered as the input to compute $\mathbf{R}_x$. Table 1 shows the FOEs estimated by locating the zero-boundaries.

Table 1. Estimation of FOEs. CA: Camera A; CB: Camera B; M1: Motion 1; M2: Motion 2.

		Ground Truth (Unit: pixel)	Experiment (Unit: pixel)
FOE	CA,M1	[29.534 2.465]	[30.000 2.950]
	CB,M1	[-5.000 30.000]	[-5.000 29.775]
	CA,M2	[30.972 9.198]	[32.000 9.475]
	CB,M2	[-3.000 27.000]	[-3.000 27.375]

Fig. 4 shows the zero-boundaries determined by the estimated FOEs exactly divided the image domain into two regions according. to the positive candidates and negative candidates.

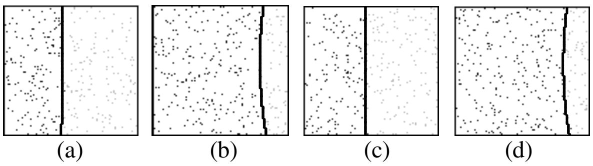


Fig. 4. The zero-boundaries (**s**=[1 0 0]) are estimated precisely. Gray dots=negative candidates; black dots = positive candidates. (a) Camera B, Motion 1; (b) Camera A, Motion 1; (c) Camera B, Motion 2; (d) Camera A, Motion 2.

Then the rotational component $\boldsymbol{\omega}_x$ of the stereo geometry was estimated from the FOEs by SVD, shown in the Tab.2.

Table 2. Estimation the rotational component of the stereo geometry. Error: 0.7964° in direction, 1.2621%in length.

	$\boldsymbol{\omega}_x$
Ground Truth	$[0.1000 \quad 0.1000 \quad -0.2000]^T$
Experiment	$[0.0974 \quad 0.2041 \quad -0.2029]^T$

Estimation of $\mathbf{t}_x$ up to Scale Using Synthetic Image Data. After the step of estimating $\mathbf{R}_x$ ($_x$), we assumed the stereo rig rotated about an axis passing the optical center of camera A at two different given velocities. Similarly, the normal flows were then generated. We also used s-axes to locate the zero boundaries on the positive-negative patters to estimate rotations of camera A $_A$ combining the algorithm named 'detranslation'[6]. And then FOE of camera B $\mathbf{t}_B$ was easily obtained from the patterns. The experimental result is shown in Table 3.

Table 3. Estimation of FOE of Camera B and rotation of Camera A. CA: Camera A; CB: Camera B; M1: Motion 1; M2: Motion 2.

		Ground Truth	Experiment
$\boldsymbol{\omega}_A$	CA,M1	$[0.8000\ 0.8000\ 8.0000]\times10^{-4}$	$[0.8023 \quad 0.8023 \quad 8.0237]\times10^{-4}$
	CA,M2	$[0.6000\ 0.4000\ 2.0000]\times10^{-4}$	$[0.5920 \quad 0.3947 \quad 1.9733]\times10^{-4}$
$\mathbf{t}_B$	CB,M1	$[0.0910 \quad 0.5495 \quad 0.0459]$	$\mathbf{t}_b/|\mathbf{t}_b|$: $[1.9645 \quad -11.2941]$
	CB,M2	$[0.0216 \quad 0.1352 \quad 0.0206]$	$\mathbf{t}_b/|\mathbf{t}_b|$: $[1.5864 \quad -6.1978]$

Finally we obtained $\mathbf{t}_x$ up to scale by using equation (3.6), as shown in Table 4.

In the experiment using synthetic data, we first assumed a motion of the cameras, and then calculated the theoretical full optical flows at each pixel. The normal flows were computed by allocating each pixel a random gradient direction. Therefore, the synthetic normal flow is consistent with its definition. Moreover, the synthetic normal

Table 4. Estimation of $\mathbf{t}_x$ up to scale. The result estimated is a unit vector describing the direction of the baseline. We compared the angle between the ground truth and the result from the synthetic experiment, and got the angle is 2.0907°.

	$\mathbf{t}_x$
Ground Truth	[-700 20 80]T
Experiment	$[\ -0.9883 \quad 0.0428 \quad 0.1466]^T$

flows are more general than the ones calculated from the image sequences, because there is no any assumption about the characteristics of the captured scene. Hence, the precision of the experimental results is able to prove the accuracy of our algorithm.

4.2 Experimental Results on Real Image Data

We also tested the proposed method using real image data. However, we only recovered $\mathbf{R}_x(\ _x)$ by setting the stereo cameras on a translational stage, due to the limitation of the equipment.

The image sequences were captured by Dragonfly CCD cameras with resolution 640×480. Some examples are shown in Fig. 5. We used the algorithm described in [7] to estimate the intrinsic parameters of the two cameras and smoothed input images by using Gaussian Filter with n=5 and σ=1.4 to eliminate the Gaussian noise.

We examined pseudo FOEs pixel by pixel within the image frames. At most 145 s-axes were applied to shrink the location of possible FOEs. The zero-boundaries determined by the estimated FOEs and specific s-axes are shown in Fig.5.

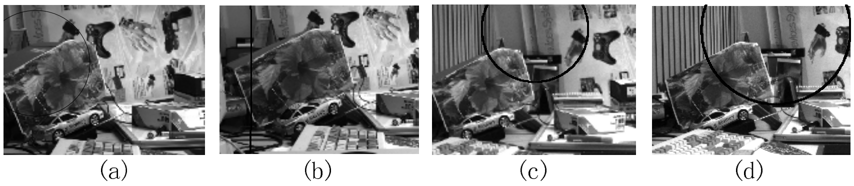


Fig. 5. The zero-boundaries determined by estimated FOEs. (a) Camera A, Motion 1, **s**=[0 0 1]; (b) Camera A, Motion 2, **s**=[1 0 0]; (c) Camera B, Motion 1, **s**=[0 0 1]; (d) Camera B, Motion 2, **s**=[0 0 1].

There are no ground truths of the extrinsic parameters of the stereo rig for us to compare our experimental result with, thus we have to compare a second individual experimental result to prove the consistency of the algorithm, so as to examine the accuracy of the results. The frame rate of the image sequences in the 2nd experiment is 1.5 times of the first experiment. The stereo geometry remained during the two experiments. Results of $\boldsymbol{\omega}_x$ in two individual experiments are shown in Table 5.

Table 5. Estimation the rotational component $\boldsymbol{\omega}_x$ of the stereo geometry. We compared the two rotational vectors and calculate the error is10.8789%in length and 1.4663° in direction.

Experiment 1	Experiment 2
$[0.0213 \quad 0.1012 \quad 0.0535]^T$	$[0.0238 \quad 0.1106 \quad 0.0622]^T$

5 Conclusion and Future Work

We have presented a new method on estimating binocular geometry directly from monocular normal flows, which requires no cross-camera correspondences nor full optical flow be accessible. Our future work is to relax the requirement of specific stereo-rig motions in the method.

Acknowledgement

The work described in this paper was partially supported by the Chinese University of Kong 2003-2004 Direct Grant (Project No. 2050298). The work is affiliated with the Microsoft-CUHK Joint Laboratory for Human-centric Computing and Interface Technologies.

References

1. M. Bjorkman, J.O. Eklundh,"Real-time epipolar geometry estimation of binocular stereo heads", *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. 24 (3), Mar. 2002.
2. F. Dornaika, R. Chung, "An algebraic approach to camera self-calibration", Computer Vision and Image Understanding, Vol.83 (3), Sept. 2001.
3. F. Dornaika, R. Chung, "Stereo geometry from 3D ego-motion streams". *IEEE Trans. On Systems, Man, and Cybernetics: Part B, Cybernetics*, Vol. 33(2) April, 2003.
4. O. Faugeras, T. luong, S. Maybank, "Camera self-calibration: theory and experiments ", *Proc. 3rd European Conf. Computer Vision*, Stockholm, Sweeden, 471-478,1994.
5. C. Fermüller and Y. Aloimonos, Direct perception of 3D motion from patterns of visual motion, *Science*, 270: 1973-1976, Dec. 1995,.
6. C. Fermüller and Y. Aloimonos, Qualitative egomotion, *Int' Journal of Computer Vision,* Vol.15, 7-29, 1995.
7. J. Heikkilä. "Geometric camera calibration using circular control points", *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 22(10), 1066-1077, Oct 2000.
8. R. Hartley, "An algorithm for self calibration from several views", *Proc. Conf. Computer Vision and Pattern Recognition*, Seattle, Washington, USA, 908-912, June 1994.
9. R. Horaud, G. Csurka, D. Demirdijian, "Stereo Calibration from Rigid Motions", *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. 22 (12), Dec. 2000.
10. K. Kanatani, Geometric Computational for Machine Vision, Clarendon Press, Oxford, 1993.
11. J. Knight, I. Reid, "Active visual alignment of a mobile stereo camera platform", *IEEE International Conference on Robotics and Automation*, pp. 3203-3208, San Francisco, CA, April, 2000.
12. S. J. Maybank and O. Faugeras, "A Theory of self-calibration of a moving camera", *Int. J. Computer Vision*, Vol. 8(2), 123-152, Aug. 1992.
13. Z. Zhang, Q.-T. Luong, and O. Faugeras, "Motion of an uncalibrated stereo rig: Self-calibration and metric reconstruction". *IEEE Trans. on Robotics and Automation*, Vol. 12 (1), 103-113, February 1996.

Singular Value Decomposition-Based Illumination Compensation in Video

Ki-Youn Lee[1] and Rae-Hong Park[1,2]

[1] Department of Electronic Engineering, Sogang University
C.P.O. Box 1142, Seoul 100-611, Korea
{amis798, rhpark}@sogang.ac.kr
[2] Interdisciplinary Program of Integrated Biotechnology, Sogang University

Abstract. This paper presents a singular value decomposition (SVD)-based illumination compensation method in video having varying scene illumination. In video that does not contain scene changes, the color distributions in the RGB space are different frame to frame, mainly due to varying illumination. In this paper, the color distribution of a scene is modeled as an ellipsoid using SVD and scene illumination of successive frames is preserved by the linear transformation in the RGB space. The effect of illumination change is effectively removed by the linear transformation and the similarity measures such as the normalized cross correlation, the sum of absolute differences, and the sum of squared differences of two successive image frames, are preserved, which illustrates the effectiveness of the proposed algorithm. Simulation results with several synthetic and real test sequences show the robustness of the proposed method to illumination changes compared with the conventional methods.

1 Introduction

Video processing algorithms, such as motion estimation and video coding, make use of spatial and temporal similarities [1]. In practice, the similarity between successive frames decreases by many factors: for example, object and/or camera motion, scene change, and illumination change between successive frames. In video containing no scene changes, the abrupt decrease of the similarity value is predominantly due to the scene illumination change. Of course, object and/or camera motion is also to be considered when scene illumination is compensated in video. When the input sequence contains (unknown) scene illumination changes between successive image frames, the performance of video processing algorithms cannot be guaranteed for a variety of video sequences with different statistical properties. Wei and Li proposed a block-matching with illumination variations (BMIV) method to remove illumination variations for motion compensation in color video [2]. Gomez-Moreno *et al.* presented a method that extracts and recovers illumination using wavelet transform for image coding [3].

The problem of obtaining constant color under different imaging condition is called color constancy and a large number of researches have been presented to solve the problem [4][5]. Forsyth [6] had proposed a gamut mapping algorithm, which was further improved by Finlayson and Hordley [7]. Forsyth showed that the shape of the color in the RGB space is a convex hull and that two images with the same surface reflectance but different shape of the gamut illustrate that lighting environment of two images is different. Linear mapping to the reference convex hull preserves lighting environment, in which linear mapping is obtained from the volume of a convex hull under the uniform and natural light. Gray world (GW) assumed that the average value of the RGB value of a scene should be gray value. This assumption is valid in general since any given real world scene has lots of different color variations [4][5]. Land and McCann proposed a retinex method [8] and Funt *et al.* implemented it in MATLAB™ [9]. Finlayson *et al.* showed that histogram equalization (HE) of each channel of a color image renders invariant color under imaging conditions (imaging device and scene illumination) [10].

In this paper, a new illumination compensation algorithm is presented, in which the color distribution of a scene in video is analyzed in the RGB color space. A scene illumination change is described by the color distribution change of a scene, in which a color distribution of a scene is modeled by an ellipsoid in the RGB color space. The illumination compensation matrix explaining the relationship between two ellipsoids is obtained using singular value decomposition (SVD), in which two ellipsoids represent color distributions of two successive frames in the RGB color space. This matrix removes the effect of illumination change between two successive frames and preserves similarity measures such as the normalized cross correlation (NCC), sum of absolute differences (SAD), and the sum of squared differences (SSD), compared with conventional methods (GW, Retinex, and HE).

2 Illumination Compensation in Video

2.1 Color Distribution Model

Gamut is a region showing defined color values in some color space and its shape in the RGB color space is represented by a convex hull [6]. In [6], gamut represents a scene illumination and gamut mapping converts an arbitrary illumination of a scene into the scene illumination under the specified illumination, e.g., daylight. Gamut mapping that makes use of the temporal redundancy is not suitable for a video because this approach considers natural light and most of surfaces are assumed to be Lambertian. Artificial and non-uniform lights having a very narrow wavelength characteristic can suddenly change illuminations, which produces a more significant change in the color distribution than the gamut change does (see Fig. 3). The color distribution in the color space gives some explanation of the color difference of two frames of the scenes. When surface reflectance and illumination in two successive frames are equal, their color distributions are the same, which does not signify that two successive frames are exactly the same on the pixel-by-pixel basis. Notice that if two images are exactly the same, their color distributions are the same. In video, the

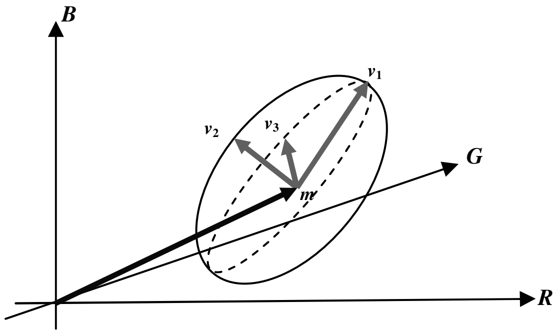


Fig. 1. An ellipsoidal color distribution model: three axes and a center of an ellipsoid represent a scene illumination. Three axes (v_1, v_2, and v_3) are obtained by SVD and a center (m) denotes an average in the RGB color space.

surface reflectance in the successive image frames is similar if a scene change does not occur, which means that a color distribution change represents the illumination change of a scene.

Fig. 1 shows a color distribution in the RGB space modeled by an ellipsoid, in which three axes and a center are illustrated. Three axes (v_1, v_2, and v_3) are obtained by SVD, representing the color distribution of a scene. The center m of an ellipsoid signifies the average of RGB values, exhibiting the average color brightness of a scene. If illumination changes exist between two successive frames, one ellipsoid is transformed (translated, rotated or distorted) into another. In cases of no distortion, in which no new objects with different surface color are inserted, the transformation between two ellipsoids is represented by the rotation and translation.

2.2 A Constraint

In modeling the color distribution by three axes of an ellipsoid, a consideration must be taken into. An axis can be equivalently expressed by two directions, for example, $(1, 1, 1)^T$ and $(-1, -1, -1)^T$. The two vectors are not the same, however, two axes represented by two vectors are exactly the same.

Fig. 2 illustrates that two ellipsoids modeled from the color distributions of two successive frames look similar, however directions of these two axes are opposite. A small shift of color around the major axis of an ellipsoid changes the direction of ellipsoid axes. Opposite direction of the axes does not guarantee the good performance of the proposed illumination compensation method. To guarantee the good performance, the sign change in the axis vector between successive frames is to be restricted. The orthonormal matrix V^T is expressed by

$$\boldsymbol{V}^T = \begin{pmatrix} v_1 & v_2 & v_3 \end{pmatrix}^T. \tag{1}$$

To remove the sign change of the axes of an ellipsoid, V^T is modified as

$$\tilde{\boldsymbol{V}}^T = \begin{pmatrix} b_1 v_1 & b_2 v_2 & b_3 v_3 \end{pmatrix}^T \tag{2}$$

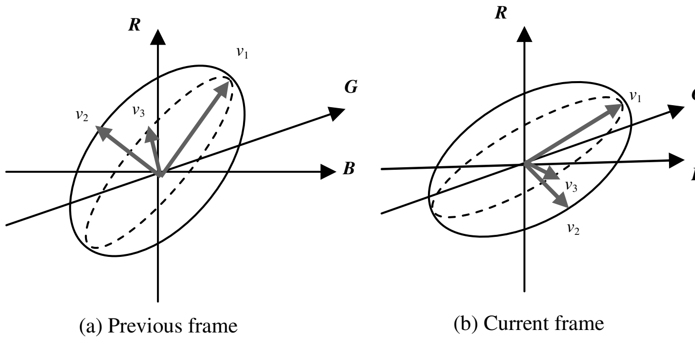


Fig. 2. Two ellipsoidal models of successive frames, in which the center of an ellipsoid is shifted to the origin of the RGB space. Two ellipsoids look similar, however the current frame shows more red color than the previous frame. The ellipsoid of the current frame inclines to the *B* axis and the directions of two axes are opposite to each other.

where b_i (i=1, 2, and 3) are binary numbers, 1 or −1 depending on the sign of v_i. If the (k+1)th frame is equal to the kth frame, b_i is set to 1. Otherwise, b_i is set to −1.

2.3 Illumination Compensation

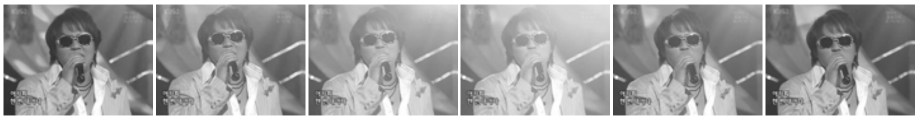
The orientation or direction of a color distribution is described by three axes of an ellipsoid and the ellipsoid center represents an average of color values in the RGB color space. A major axis and the other two orthonormal axes are obtained by SVD [11]. Let $\boldsymbol{A}$ be a $P\times3$ matrix, where P represents the number of image pixels in each of three RGB color channels. Then, $\boldsymbol{A}$ is decomposed by

$$\boldsymbol{A}-\boldsymbol{O}\,\boldsymbol{m}^T=\boldsymbol{U}\,\boldsymbol{\Sigma}\,\boldsymbol{V}^T \tag{3}$$

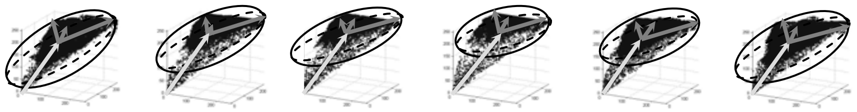
where represents a singular value matrix, $\boldsymbol{V}^T$ signifies an orthonormal matrix, $\boldsymbol{m}$ denotes the average column vector of columns of $\boldsymbol{A}$, and $\boldsymbol{O}$ is a $P\times1$ column vector with all elements being equal to one. Fig. 1 shows the ellipsoid model of the color distribution in the RGB space, where three column vectors (v_1, v_2, and v_3) are orthonormal and represent the rotation of an ellipsoid. A column vector of $\boldsymbol{V}^T$ with the largest singular value of denotes the major axis of an ellipsoid. If $\boldsymbol{V}^T$ of the current frame is quite different from that of the next frame, the illuminations of two successive frames are not the same.

Fig. 3 shows six successive frames, which contain object motions and illumination changes, as well as their color distributions. Note that the gamut of six successive frames is similar although artificial and non-uniform light is incident from top right. But the color distributions of six successive frames are quite different. When blue and green light is incident at top right, *G* and *B* values at each pixel increase. Note that only the pixels affected by the illumination change have the increased *G* and *B* color values. This phenomenon changes the color distribution, which is shown in Fig. 3(b).

To preserve the same scene illumination, three directions and a center of the ellipsoid that represent the color distribution have to be the same for successive image



(a) Real video sequences



(b) Their color distributions

Fig. 3. Six successive image frames (a) and their color distributions in the RGB color space (b). Blue and some green lights suddenly illuminate at top right at frame 21. The illumination changes object and background colors, making color at top right more bluish and greenish. In this test video containing small object motions, the illumination change mainly produces the color distribution shift in the RGB color space.

frames, preserving the similarity value between two successive frames of the same scene. With a constraint, the *k*th frame is decomposed by

$$\boldsymbol{A}_k - \boldsymbol{O}\,\boldsymbol{m}_k^T = \boldsymbol{U}_k\,\boldsymbol{\Sigma}_k\,\tilde{\boldsymbol{V}}_k^T \tag{4}$$

and that of the (*k*+1)th frame is decomposed by

$$\boldsymbol{A}_{k+1} - \boldsymbol{O}\,\boldsymbol{m}_{k+1}^T = \boldsymbol{U}_{k+1}\,\boldsymbol{\Sigma}_{k+1}\,\tilde{\boldsymbol{V}}_{k+1}^T\,. \tag{5}$$

Then, the illumination compensation matrix $\boldsymbol{T}_{k+1,\,k}$ is given by

$$\tilde{\boldsymbol{V}}_k^T = \tilde{\boldsymbol{V}}_{k+1}^T \boldsymbol{T}_{k+1,k}\,. \tag{6}$$

The illumination-compensated image $\boldsymbol{A}'_{k+1}$ is computed as

$$\boldsymbol{A}'_{k+1} = \left(A_{k+1} - \boldsymbol{O}\,\boldsymbol{m}_{k+1}^T\right)\boldsymbol{T}_{k+1,k} + \boldsymbol{O}\,\boldsymbol{m}_{k+1}^T \tag{7}$$

where $\boldsymbol{m}_k$ and $\boldsymbol{m}_{k+1}$ denote the average RGB values of the *k*th and (*k*+1)th frames, respectively. Three directions and a center of $\boldsymbol{A}'_{k+1}$ should be equal to those of $\boldsymbol{A}_k$, making the illuminations of successive image frames equal.

3 Experimental Results and Discussions

In the previous section, it is assumed that the illumination change between successive frames is estimated by the change of a color distribution in the RGB space and then removed by the compensation of a color distribution using SVD. To show the effectiveness of the proposed scene illumination compensation method using SVD, first of all, the GretagMacbeth color checker chart (24c) illuminated under the

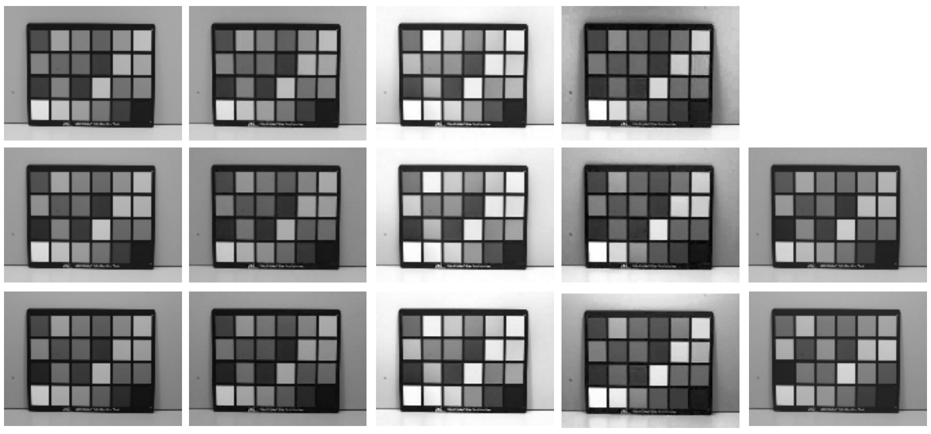
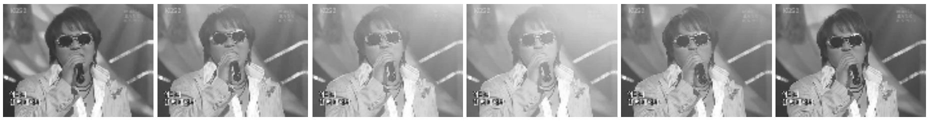
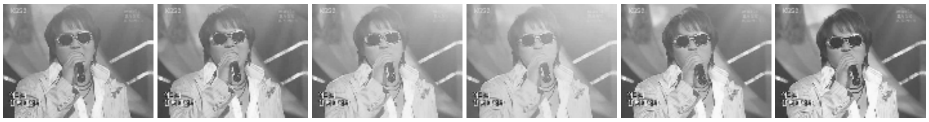


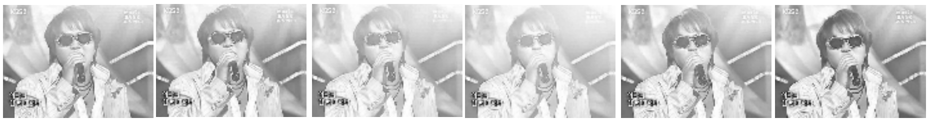
Fig. 4. GretagMacbeth 24 color charts under representative illuminations (a). ((b)-(d)) represent the illumination-compensated images of (a) using the conventional methods. Illumination-compensated images by the proposed algorithm are shown in (e).

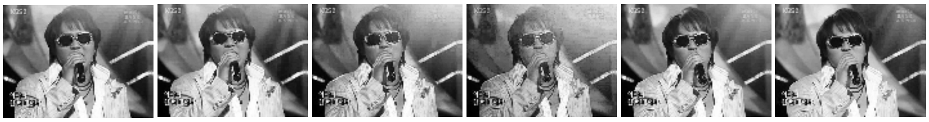
representative illuminations is used as a test image. Fig. 4(a) shows the GretagMacbeth color checker chart (24c) under three representative illuminations: 6500K, 4156K, and 2856K from top to bottom. Their images are captured by Sony 3CCD camera (DXC-990) in the lighting device (Spectralight III). Figs. 4(b), 4(c), and 4(d) show illumination-compensated images by the conventional methods (Gray world (GW), Retinex and histogram equalization (HE), respectively). In experiments of the proposed algorithm, the image under 6500K (at the top of Fig. 4(a)) is chosen as the reference image and Fig. 4(e) illustrates illumination-compensated images by the proposed method, under the same scene illumination as in Fig. 4(a), showing that their scene illuminations are compensated better than scene illuminations of the conventional methods. The proposed method converts color checker chart (24c) images under 4156K and 2856K into those under 6500K (note that the top image is empty in Fig. 4(e)).

Next, we use a real video sequence that is composed of 48 frames. Fig. 3(a) shows six successive frames containing object motions and illumination changes. Varying illumination shifts the color values in the RGB color space to the direction of the color of illumination. Note that an occlusion of surfaces more or less affects the color distribution. In successive frames, most color values changed by the illumination changes are shifted to the direction of the illumination color in the color space. An illumination change, not just an illumination itself, is estimated by this shift in the RGB space. We can obtain the information of this shift using SVD and the average of RGB values. Fig. 5 shows six successive illumination-compensated image frames by the conventional methods and the proposed method. The proposed method effectively removes some green illuminations at top right. Conventional methods (GW, Retinex, and HE) cannot effectively remove them. Note that the synthetic color value is fixed with varying lighting condition and does not follow the model mentioned in the previous section, in which the model assumes that the color values


(a) Original real image sequences


(b) GW


(c) Retinex


(d) HE

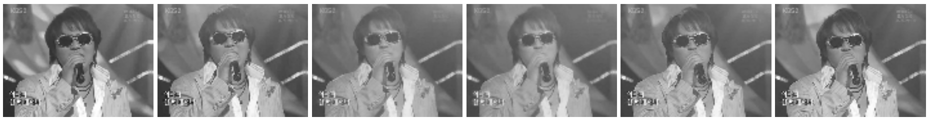

(e) Proposed

Fig. 5. Original real image sequences (a) and illumination-compensated image frames by the conventional methods ((b)-(d)) and the proposed method (e)

are shifted by the change of scene illumination and the change of scene illumination can be estimated from the change of the color distribution in the RGB color space.

The previous section explains a constraint, i.e., the signs of vectors of two ellipsoids, which represent the color distribution directions of two successive frames, must be the same, in the context of the geometrical model in which the color distribution is modeled by an ellipsoid. Two opposite directions by the sign change of vectors are considered to represent the same direction. Figs. 6(b) and 6(c) show that the constraint must be considered for better results. From frame 0 to frame 35, the illumination compensation is acceptable in terms of the NCC, and thus the constraint is not needed. From frame 36 to frame 48, directions of color distributions look similar, however the directions of v_2 and v_3 or the directions of v_3 of two ellipsoids, estimated from two successive frames, are opposite. In that case, a constraint is to be applied (i.e., by removing the sign change between two successive frames) to ensure the performance of the proposed illumination compensation method.

Fig. 7 shows the similarity measure as a function of the frame index, in terms of the NCC, of real image frames and illumination-compensated image frames. Fig. 7(a) illustrates the NCC of the original sequences and Fig. 7(e) shows the NCC of the illumination-compensated sequence by the proposed algorithm. Figs. 7(b), 7(c), and

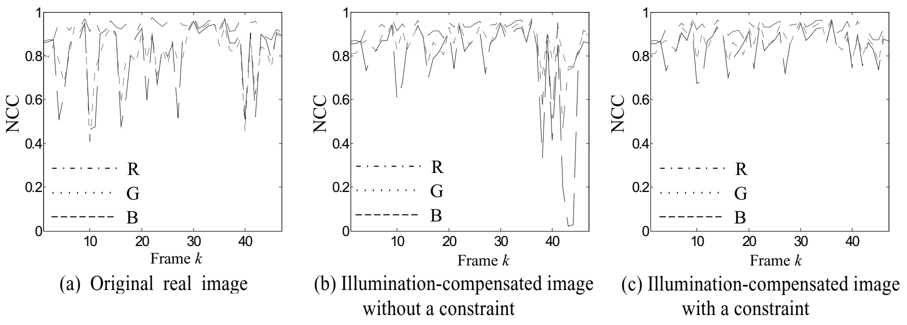


Fig. 6. NCC, as a function of the frame index of original real image sequences (a), of the illumination-compensated image sequence by the proposed method without (b) and with (c) a constraint

7(d) illustrate the NCC of the illumination-compensated sequences by the conventional GW, Retinex, and HE, respectively.

NCC values of the input (original) image sequence are small at frames at which abrupt illumination change occurs. Because illumination color is blue and is mixed with some green, the NCC values of *B* and *G* channels abruptly decrease (Fig. 7(a)). NCC values in *B* and *G* channels of the illumination-compensated image sequences

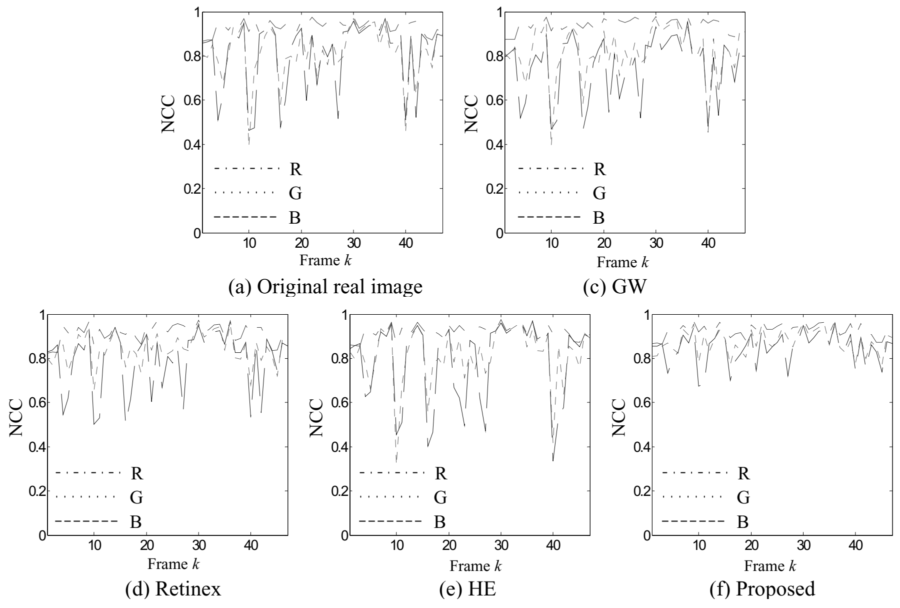


Fig. 7. NCC, as a function of the frame index of original real image sequences (a), of the illumination-compensated image sequence by GW (b), Retinex (c) and HE (d), of the illumination-compensated image sequence by the proposed method (e)

Table 1. Performance comparison of the proposed method with the conventional methods

	color checker charts (24c)	Real video			Remark (Reconstruction)
		NCC	SAD	SSD	
GW	○	Δ	Δ	Δ	○
Retinex	○	Δ	○	○	×
HE	○	×	×	×	Δ
Proposed	○	○	○	Δ	○

increase (Fig. 7(e)), preserving the similarity measure even in video images with varying scene illumination. Figs. 7(b)-7(d) show that NCC values of illumination-compensated image sequences are not increased much by the conventional methods (GW, Retinex, and HE).

Table 1 shows the performance of similarity measures such as the NCC, SAD, and SSD, in which '○' represents effective improvement of the performance, 'Δ' signifies slight improvement, and '×' denotes no improvement. The retinex method and the proposed method effectively improve the performance in terms of the specified similarity measures. It is shown that illumination-compensation methods require a number of coefficients, which make the compensated image sequences equal to the original image sequences. GW and the proposed methods require 3 and 12 coefficients to reconstruct the original image sequence in each frame. HE requires the cumulative density function to recover the original image sequence. It is difficult to recover the original image sequences by illumination compensation by a retinex algorithm. If the difference image between the original image and illumination-compensated image is given, the original image sequences can be recovered.

Especially for video compression, the proposed method has several advantages: it works well in image sequences containing object and/or camera motions. Illumination compensation is performed using only 12 coefficients (two types of parameters: 3×3 illumination compensation matrix and the means of RGB channels). Compensated-image sequences are easily recovered to the original image sequences.

Table 2 shows the performance comparison of the proposed method and the conventional methods in terms of the computation time, in which 48 frames of 320 × 240 images are used. The codes of four methods (GW, Retinex, HE, and the proposed method) are programmed in MATLAB on 3.2 GHz CPU processor. The proposed method has the highest computational complexity mainly due to the time taken to obtain three orthonormal vectors using SVD. The proposed method is suitable for applications in which encoding time is less critical than decoding time.

Table 2. Performance comparison of the proposed method and the conventional methods in terms of the computation time (uint: second)

	GW	Retinex	HE	Proposed
Computation time	2.61	226.75	2.84	871.47

4 Conclusions

This paper proposes an illumination compensation method in video robust to varying scene illumination. In the proposed algorithm, the change of scene illumination is represented by the change of color distribution that is modeled by an ellipsoid using SVD, in which three axes and a center of an ellipsoid are used. Simple computation (linear transformation with a constraint) is enough for removing illumination variations. Similarity measures such as the NCC, SAD, and SSD are preserved by the proposed illumination compensation method. Synthetic images and real video image frames are used to show the effectiveness of the proposed illumination compensation method for images containing illumination changes. The effectiveness of the proposed algorithm is also shown in terms of the subjective quality of the illumination-compensated image frames and the quantitative similarity measure such as the NCC, SAD, and SSD.

Acknowledgement

This work was supported by Brain Korea 21 project.

References

1. Wang, Y., Ostermann, J., Zhang. Y. Q.: Video Processing and Communications. Prentice-Hall (2002)
2. Wei, J., Li, Z. N.: Motion Compensation in Color Video with Illumination Variations. Proc. IEEE Conf. Image Processing, Vol. 3 (1997) 614−617
3. Gomez-Moreno, H., Maldonado-Bascon, S., Lopez-Ferreras, F., Acevedo-Rodriguez, F. J.: Extracting Illumination from Images by Using the Wavelet Transform, Proc. IEEE Conf. Image Processing, Vol. 2 (2001) 265−268
4. Barnard, K., Cardei, V., Funt, B.: A Comparison of Computational Color Constancy Algorithms-Part I: Methodology and Experiments with Synthesized Data. IEEE Trans. Image Processing, Vol. 11 (2002) 972−984
5. Barnard, K., Cardei, V., Funt, B.: A Comparison of Computational Color Constancy Algorithms-Part II: Experiments with Image Data. IEEE Trans. Image Processing, Vol. 11 (2002) 985−996
6. Forsyth, D. A.: A Novel Algorithm for Color Constancy. Int. J. Computer Vision, Vol. 1 (1990) 5−36
7. Finlayson, G., Hordley, S.: Improving Gamut Mapping Color Constancy. IEEE Trans. Image Processing, Vol. 9 (2000) 1774−1783
8. Land, E., McCann, J.: Lightness and Retinex Theory. J. Opt. Soc. Am. 61 (1971) 1−11
9. Funt, B., Ciurea, F., McCann, J.: Retinex in MATLAB^{TM}. J. Electronic Imaging, Vol. 13 (2004) 48−57
10. Finlayson, G., Hordley, S., Schaefer, G., Tian, G. Y.: Illumination and Device Invariant Colour using Histogram Equalisation. Pattern Recognition, Vol. 38 (2005) 179−190
11. Strang, G.: Introduction to Linear Algebra. Wellesley Cambridge Press (2003)

Facial Expression Transformations for Expression-Invariant Face Recognition

Hyung-Soo Lee and Daijin Kim

Department of Computer Science and Engineering
Pohang University of Science and Technology,
{sooz, dkim}@postech.ac.kr

Abstract. This paper presents a method of expression-invariant face recognition by transforming the input face image with an arbitrary expression into its corresponding neutral facial expression image. When a new face image with an arbitrary expression is queried, it is represented by a feature vector using the active appearance model. Then, the facial expression state of the queried feature vector is identified by the facial expression recognizer. Then, the queried feature vector is transformed into the neutral facial expression vector using the identified facial expression state via the *direct* or *indirect* facial expression transformation, where the former uses the bilinear translation directly to transform the facial expression, but the latter uses the bilinear translation to obtain the relative expression parameters and transforms the facial expression indirectly by the obtained relative expression parameters. Then, the neutral facial expression vector is converted into the neutral facial expression image via the AAM reconstruction. Finally, the face recognition has been performed by the distance-based matching technique. Experimental results show that the proposed expression-invariant face recognition method is very robust under a variety of facial expressions.

1 Introduction

Since Kanade[1] attempted an automatical face recognition 30 years ago, many researchers have investigated the face recognition. Among various face recognition methods, holistic appearance-based approach seems to have been prevailed up to now. However, face appearances are varied drastically with changes of poses, illuminations, facial expressions, and so forth. Such variations make the appearance-based face recognition difficult. This paper attempts to overcome the variations of facial expression among many difficulties.

There were many previous researches to recognize individuals across different expressions. Liu et al.[2] measured two types of the facial asymmetric information, density difference and edge orientation. They showed that this facial asymmetric information could obtain individual differences which are stable to the changes of facial expressions. Elad and Kimmel[3] proposed an efficient isometric transformation framework of non-rigid object on the manifold. This framework overcame the disadvantage of taking the rigid transformation of existing isometric transformation by using the multidimensional scaling (MDS). Wang and

Ahuja[4] decomposed facial expression features using the higher-order singular value decomposition (HOSVD) on the expression subspace and performed face recognition and facial expression recognition at the same time in the subspace. On the other hand, some researchers treated face geometry and texture information separately. The separate modeling of geometry and texture information has been occurred from the active appearance model (AAM). Li et al.[5] used the AAM to recognize individuals with face expression. They were fitting AAM to the input image and warping to the reference image frame to remove the geometry information. However, the texture information still included expression features even though their approach was better than the others.

In this paper, we transform the input arbitrary facial expression image into its corresponding neutral facial expression image for expression-invariant face recognition, assuming that gallery images consist of neutral facial expression images. Bilinear model is an appropriate technique for synthesizing new expression image from input expression image with unchanged identity. Abboud and Davoine[6] used AAM and bilinear model to synthesize expression images. However the identity of synthesized image by their method is far different from the ground-truth image, thus their method is not appropriate for face recognition. We adopted the result of Zhou and Lin[7] to synthesize realistic facial expression images and extended their work to the face recognition application.

First, we extract facial feature vector from the input image using AAM. Then, we obtain the facial expression state of the input facial feature vector by the facial expression recognizer. Then, we transform the input facial feature vector into its corresponding neutral facial expression vector using the direct or indirect facial expression transformation, and convert the neutral facial expression vector into the neutral facial expression image via the AAM reconstruction. Finally, we perform the expression-invariant face recognition by the distance-based matching techniques like nearest neighbor classifier, linear discriminant analysis(LDA), and generalized discriminant analysis (GDA).

2 Facial Feature Extraction Using AAM

The 2D shape of an AAM is defined by a 2D triangulated mesh and in particular the vertex locations of the mesh. AAMs are normally computed from training data consisting of a set of images with the shape mesh (usually hand) marked on them. The Principal Component Analysis (PCA) is then applied to the training meshes. The base shape $\mathbf{s_0}$ is the mean shape and the matrices s_i are the (reshaped) eigenvectors corresponding to the m largest eigenvalues. The 2D appearance is defined in the mean shape $\mathbf{s_0}$ and its variation is modelled by the linear combination of a mean 2D appearance A_0 and orthogonal 2D appearance bases A_i. Finally, to control both the shape and 2D appearance of the model, we generate the concatenated vector $\mathbf{b}$ which is the combination of weighted shape vector and 2D appearance vector

$$\mathbf{b} = \begin{pmatrix} \mathbf{\Psi}_s p_i \\ \alpha_i \end{pmatrix}, \tag{1}$$

where $\boldsymbol{\Psi}_s$ is a diagonal matrix of weights for each shape parameter. We apply a further PCA on vector $\mathbf{b}$ to get the feature vector $\mathbf{y}$. We use the feature vector $\mathbf{y}$ as the input of bilinear model in the following sections.

3 Bilinear Model

Bilinear model[8] is a two-factor model that separates the observation into the style and the content. In this paper, we set the facial identity as a content factor and the facial expression as a style factor. When new face image is appeared, we can change the facial expression of the face image, while the identity is fixed. Bilinear model is categorized as symmetric model and asymmetric model. Since the symmetric bilinear model tries to estimate the identity factor and expression factor simultaneously, it often fails to decompose the input image into two factors. To get a robust face recognition, we adopt the asymmetric bilinear model in this work.

Asymmetric bilinear model can be expressed as

$$\mathbf{y} = \sum_{j=1}^{J} \left(\sum_{i=1}^{I} \mathbf{w}_{ij} a_i \right) b_j, = \mathbf{W}_s \mathbf{b}, \tag{2}$$

where $\mathbf{W}_s$ is a style specific basis matrix with a known style s, $\mathbf{b}$ is a content parameter vector, J is the dimension of the content vector, and I is the dimension of the style vector, respectively. Assume that we have S(Style) $\times$ C(Content) training samples and built the observation matrix $\mathbf{Y}$ by stacking them. Then, the asymmetric model can be rewritten in a compact form as

$$\mathbf{Y} = \mathbf{W}\mathbf{B}, \tag{3}$$

where $\mathbf{W}$ and $\mathbf{B}$ have the sizes of $SK \times J$ and $J \times C$, respectively.(K is the dimension of observation vector.)

Usually, the optimal content matrix $\mathbf{B}$ and the interaction matrix $\mathbf{W}$ are estimated by applying the singular value decomposition of the observation matrix $\mathbf{Y}$. From the SVD of $\mathbf{Y}$, we obtain the decomposition result $\mathbf{Y} = \mathbf{U}\mathbf{S}\mathbf{V}^T$. Then, $\mathbf{W}$ and $\mathbf{B}$ are obtained from the first J columns of $\mathbf{US}$ and the first J rows of $\mathbf{V}^T$, respectively.

When an observation vector $\mathbf{y}$ with known style s is appeared, the content vector $\mathbf{b}$ can be computed by a single pseudo inverse operation as

$$\mathbf{b} = (\mathbf{W}_s)^{\dagger} \mathbf{y}, \tag{4}$$

where the symbol $\dagger$ denotes the pseudo inverse operation. After obtaining content factor $\mathbf{b}$, we can synthesize new feature vector with the same content and different style n as $\mathbf{y}' = \mathbf{W}_n \mathbf{b}$. The readers is referred to [8] for details.

In addition, we proposed ridge regressive bilinear model[9] which combines the ridge regression into the bilinear model. This combination provides some advantages: it makes the bilinear model more stable by shrinking the range of content and style factors appropriately, and it improves the recognition performance. In the case of asymmetric ridge regressive bilinear model, the content vector can be computed by the following simple equation:

$$\mathbf{b} = \left(\mathbf{W}^T\mathbf{W} + \lambda\mathbf{I}\right)^{-1}\mathbf{W}^T\mathbf{y} \tag{5}$$

4 Expression-Invariant Face Recognition

For the expression-invariant face recognition, we assume that the face images in the gallery have only neutral facial expressions. Then, a new face image with an arbitrary facial expression is queried, it is transformed into its corresponding neutral facial expression image. We adopt asymmetric bilinear model for this transformation and treat the facial identity and the facial expression as a content factor and a style factor of bilinear model, respectively. We assume that the style factor (i.e. facial expression) of the input face image is already known in virtue of the facial expression recognition. We adopt a double layered GDA-based facial expression recognizer[10]. Then, we transform the input face image with a known facial expression into its corresponding neutral facial expression image in terms of the direct or indirect facial expression transformation which will be explained later. Then, we perform the expression-invariant face recognition using the transformed neutral facial expression images.

4.1 Direct Facial Expression Transformation

Assume that the style factor s (facial expression) of the input face image is known by the facial expression recognizer. The output of facial expression recognizer provides the facial expression state s, which is one of the facial expressions: Happy, Surprise, Anger, Disgust, Sad, and Fear. Direct method transforms the input face image with an arbitrary facial expression into its corresponding neutral facial expression image as follows.

First, we perform the AAM fitting for the input image I and extract the facial feature vector $\mathbf{y}$ by the method described in Section 2. Second, we obtain the style s (facial expression) of the input image by the facial expression recognizer. Third, we compute the content (identity) vector $\mathbf{b}_{iden}$ by a simple operation like Eq. 4 or 5 using the known expression-specific basis matrix $\mathbf{W}_s$, where the style factor s can be known by the facial expression recognizer. Fourth, we translate the facial expression vector $\mathbf{y}$ with a facial expression s into its corresponding neutral facial expression vector $\mathbf{y}'$ by multiplying neutral expression-specific basis matrix $\mathbf{W}_n$ with the identity factor $\mathbf{b}_{iden}$. Finally, we obtain the neutral facial expression image I' by reconstructing AAM parameters of $\mathbf{y}'$. Fig. 1 summarizes an overall procedure that transforms the input face image with a specific facial expression into its corresponding neutral facial expression image.

Fig. 3-(a) shows some examples of the transformed facial images, where each column represents five different subjects and each rows represent the input surprise facial expression images (row 1), the bilinear model fitted images (row 2),

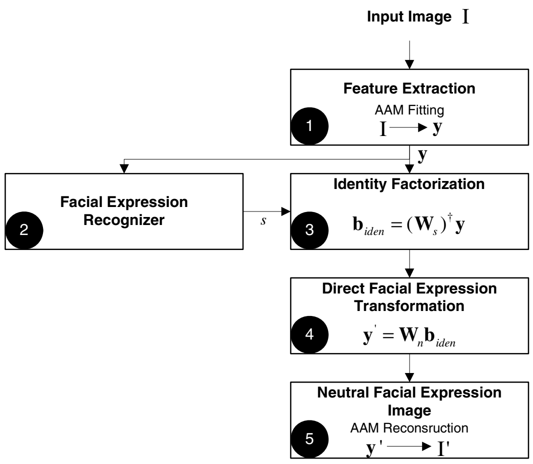

Input Image I
Feature Extraction
AAM Fitting
I → y
1
y
Facial Expression Recognizer
2
s
Identity Factorization
$\mathbf{b}_{iden} = (\mathbf{W}_s)^{\dagger}\mathbf{y}$
3
Direct Facial Expression Transformation
$\mathbf{y}' = \mathbf{W}_n \mathbf{b}_{iden}$
4
Neutral Facial Expression Image
AAM Reconsruction
y' → I'
5

Fig. 1. An overall procedure of the direct facial expression transformation

the neutral facial expression images using the direct facial expression transformation (row 3), the neutral facial expression images using the indirect facial expression transformation (row 4) and the ground-truth neutral facial expression images (row 5) respectively.

As you can see, we successfully transform the input surprise facial expression images (row 1) into the neutral facial expression images (row 3). Here, if the transformation is perfect, the transformed images (row 3) must be almost identical with the ground-truth images (row 5). However, the identities of transformed neutral facial expression images (row 3) are far from those of the ground-truth face images (row 5). This is mainly because we have a small number of subjects in our training set and the bilinear model trained with a limited number of training set can not fully express new face images far different from the training data set and accordingly the identities of the transformed face images are also far from those of input face images. As a result, the performance of face recognition using the direct facial expression transformation is poor. Therefore, we need a novel facial expression transformation method that can well express a new subject who is not contained in the training set.

4.2 Indirect Facial Expression Transformation

As mentioned before, the identity discrepancy between the transformed images and the ground-truth images comes from the fact the bilinear model itself can not express a new person who is not contained in the training set. To avoid this problem, we take the indirect facial expression transformation that performs the bilinear translation to obtain the relative expression parameters and transforms the facial expression indirectly by the obtained relative expression parameters.

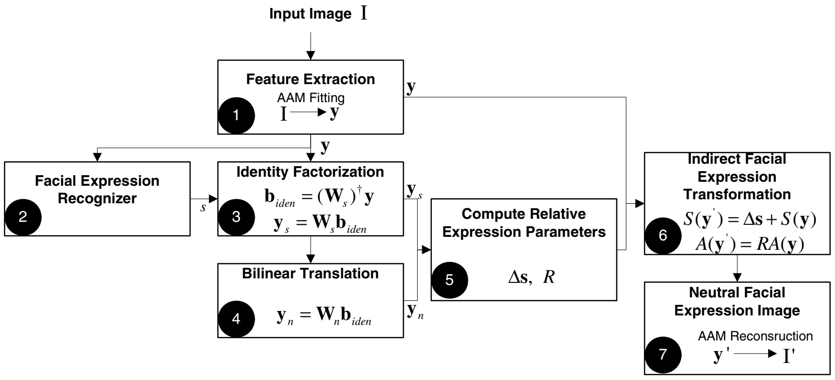
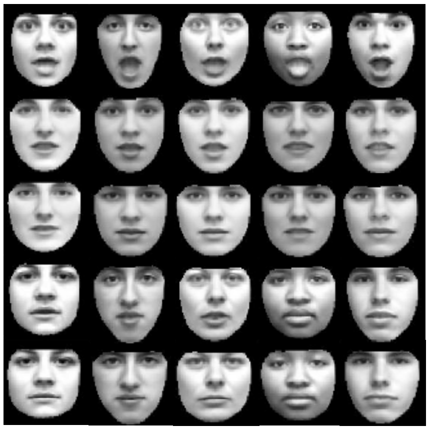

Input Image I
Feature Extraction
AAM Fitting
I → y
y
Facial Expression Recognizer
s
Identity Factorization
$\mathbf{b}_{iden} = (\mathbf{W}_s)^{\dagger}\mathbf{y}$
$\mathbf{y}_s = \mathbf{W}_s\mathbf{b}_{iden}$
$\mathbf{y}_s$
Bilinear Translation
$\mathbf{y}_n = \mathbf{W}_n\mathbf{b}_{iden}$
$\mathbf{y}_n$
Compute Relative Expression Parameters
Δs, R
Indirect Facial Expression Transformation
$S(\mathbf{y}') = \Delta\mathbf{s} + S(\mathbf{y})$
$A(\mathbf{y}') = RA(\mathbf{y})$
Neutral Facial Expression Image
AAM Reconsruction
y' → I'
1
2
3
4
5
6
7

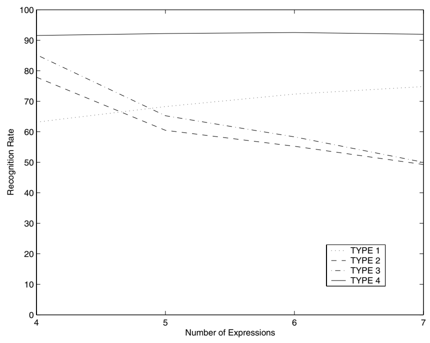
Fig. 2. An overall procedure of the indirect facial expression transformation

Zhou and Lin[7] proposed the relative expression parameters such as the shape difference and the appearance ratio in order to get the robust facial expression image synthesis. The basic idea of their approach can be described as follows. Let $\Delta\mathbf{s} = \mathbf{s}_n - \mathbf{s}_s$ be the shape difference between the feature points of neutral facial expression face image and that of a specific facial expression face image. Then, if $\mathbf{s}_s^{'}$ is the shape of a specific facial expression image of a new person, the transformed neutral facial expression image $\mathbf{s}_n^{'}$ can be obtained by $\mathbf{s}_n^{'} = \mathbf{s}_s^{'} + \Delta\mathbf{s}$. If we assume that two persons with the same expression are in the same pose, lighting condition, and their face image are warped to the same shape, the appearance ratio of two persons which defined as $R(u,v) = \frac{A_n(u,v)}{A_s(u,v)}$ will be almost the same, where (u,v) are the coordinates of a pixel in the images, A_n and A_s are the appearances of the neutral and a specific expression face image, respectively. Then, if A_s' is the appearance with a specific facial expression s of a new person, the transformed neutral facial appearance A_n' of that person can be obtained by $A_n'(u,v) = R(u,v)A_s'(u,v)$.

Fig. 2 summarizes an overall procedure of the indirect facial expression transformation of a new person by using relative expression parameters (shape difference and appearance ratio). The detailed explanation of the procedure is given below.

1. Perform the AAM fitting for the input image I and extract facial feature vector $\mathbf{y}$ by the method described in Section 2.
2. Perform a facial expression recognition and obtain the facial expression state s.
3. Obtain a style specific basis matrix $\mathbf{W}_s$. Then, compute the content (identity) vector $\mathbf{b}_{iden}$ and obtain the bilinear model fitted feature vector $\mathbf{y}_s = \mathbf{W}_s\mathbf{b}_{iden}$.
4. Obtain the neutral facial expression feature vector $\mathbf{y}_n = \mathbf{W}_n\mathbf{b}_{iden}$.
5. Compute the relative expression parameters such as shape difference and appearance ratio as $\Delta\mathbf{s} = \mathbf{s}_{\mathbf{y}_n} - \mathbf{s}_{\mathbf{y}_s}$, $R(u,v) = \frac{A_{\mathbf{y}_n}(u,v)}{A_{\mathbf{y}_s}(u,v)}$, where $\mathbf{s}_{\mathbf{y}_n}$ and $\mathbf{s}_{\mathbf{y}_s}$ are the shape vectors of $\mathbf{y}_n$ and $\mathbf{y}_s$, $A_{\mathbf{y}_n}$ and $A_{\mathbf{y}_s}$ are the appearance images of $\mathbf{y}_n$ and $\mathbf{y}_s$, and (u,v) is the 2D coordinates of a pixel in the appearance image.


(a) Some facial expression images obtained from the direct and indirect facial expression transformation



(b) The face recognition rate vs the different numbers of facial expressions

Fig. 3. The experimental results

6. Transform the input feature vector $\mathbf{y}$ into its corresponding neutral facial expression feature vector $\mathbf{y}^{'}$ using the relative expression parameters $\Delta\mathbf{s}$ and R as $\mathbf{s}_{\mathbf{y}'} = \Delta\mathbf{s} + \mathbf{s}_{\mathbf{y}}, \quad A_{\mathbf{y}'}(u,v) = R(u,v)A_{\mathbf{y}}(u,v)$.
7. Obtain the neutral facial expression image I' by reconstructing AAM parameters of the neutral facial expression feature vector $\mathbf{y}'$.

As you can see from fig. 3-(a), the identities and shapes of the neutral facial expression images using indirect facial expression transformation (row 4) are almost identical with those of the ground-truth images (row 5). It is mainly because the neutral facial expression images are obtained from a combination of the trained bases of bilinear model in the direct facial expression transformation, but they are obtained by modifying the input images by the relative expression parameters in the indirect facial expression transformation. Thus, the indirect facial expression transformation provides a more effective facial expression transformation of a new person who is not contained in the training data set and this improves the performance of facial recognition greatly.

5 Experimental Results and Discussion

5.1 Experiment Setup

To evaluate the proposed expression-invariant face recognition method, we used "Cohn-Kanade AU-Coded Facial Expression Database" [11] for the following experiments. The database contains seven facial expressions: neutral, happy, surprise, anger, disgust, sad and fear. Each subject in the database has at most seven facial expressions, i.e. not all the subject has seven facial expressions due to the incompleteness of the database. We manually selected 1,020 frontal face images from the image sequences. With the selected face images, we construct the active appearance model for facial feature extraction. The constructed model is built using 56 shape bases, 79 2D appearance bases and 104 concatenated feature bases. Each number of bases is selected to account for 95% shape, appearance, and feature variation. Thus, the observation vector $\mathbf{y}$ for the bilinear model is a 104 dimensional vector.

First, we extracted the facial features from all face images in the database using the AAM. Then, we divided the face database into the training set and the test set. With the training set, we performed asymmetric bilinear model learning and obtained the expression specific basis matrices $\mathbf{W}_s$, $s = 1, \cdots, 7$, where s denotes a specific facial expression among seven different facial expressions. We divided the test set into the gallery set and the probe set. The gallery set consists of neutral facial expression images of all subject in the test set, while the probe set consists of all remaining facial expression images of all subjects in the test set. Then, we performed the facial expression recognition using the feature vector of the probe image and determined the style factor (facial expression). Then, we obtained the identity factor of the probe image by multiplying the pseudo inverse of the expression specific basis matrix and the feature vector of the probe image. Then, we transformed the probe image into its corresponding neutral expression image by the transformation method I or II. Finally, the identity of the gallery image which has the minimum Euclidean distance to transformed probe image is selected as the identity of the probe image.

We performed the experiments with several different conditions by changing the number of facial expressions. Table 1 summarizes the data configuration and the parameter values for each experiment. For example, for the experiment with 7 expressions, we divided 1,020 facial images of 95 subjects into two data sets: training set and test set as follows. Among 1,020 facial images, only 11 subjects have all 7 facial expressions. Thus, the 198 facial images of 11 subjects are taken for training the bilinear model and the remaining 822 images of 84 subjects are taken for testing the recognition performance. The training and test data set in the other experiments are taken in a similar manner. We built the observation matrix $\mathbf{Y}$ by stacking the feature vector $\mathbf{y}$ of each training data. Each column of $\mathbf{Y}$ has the feature vector $\mathbf{y}$ of a specific subject with all expressions and each row has the feature vector of all the subjects for a specific expression. We take $S = 7$, $K = 104$, $J = 11$ and $C = 11$ for Eq. (3) and take $\lambda = 8$ for ridge regressive bilinear model, which showed the best classification performance through many different trials of experiments.

5.2 The Effect of the Number of Training Subjects

We performed four different types of facial expression transformations: the face recognition with no facial expression transformation (TYPE 1), the face recognition with the direct facial expression transformation using the bilinear model (TYPE 2), the face recognition with the direct facial expression transformation using the ridge regressive bilinear model (TYPE 3) and the face recognition with the indirect facial expression transformation (TYPE 4). We performed these experiments with four different numbers of facial expressions: 4, 5, 6 and 7, and

Table 1. Experimental configuration

Number of expressions	Number of training subjects	Number of test subjects	S	C	J	λ
4	38	57	4	38	38	17
5	21	74	5	21	21	13
6	14	81	6	14	14	10
7	11	84	7	11	11	8

Table 2. Face recognition results with 4 facial expressions

Transformations	Methods	Happy	Surprise	Fear	Average
TYPE 1	NN	81.82	28.95	100	63.16
	LDA+NN	95.45	71.05	84.61	**84.21**
	GDA+NN	90.9	71.05	92.3	83.15
TYPE 2	NN	84.09	68.42	84.61	77.89
	LDA+NN	86.36	86.84	84.61	**86.31**
	GDA+NN	97.72	78.94	61.53	85.26
TYPE 3	NN	93.18	76.31	82	85.26
	LDA+NN	93.18	89.47	90	**92.63**
	GDA+NN	90.9	94.73	94	**92.63**
TYPE 4	NN	95.45	84.21	100	91.57
	LDA+NN	90.9	97.36	100	94.73
	GDA+NN	97.72	94.73	100	**96.84**

the numbers of training subjects for each experiment are 38, 21, 14, and 11, respectively. Fig. 3-(b) shows that (1) the face recognition rate using TYPE 1 rather increases as the number of facial expressions increases. This is because the facial expressions such as anger, disgust and sad are similar with the neutral facial expression and the addition of these facial expressions increases the average recognition rate; (2) the face recognition rate using TYPE 2 or TYPE 3 decreases as the number of facial expressions increases. This is because the number of training subjects decreases as the number of facial expressions increases and it makes the transformed probe images far different from gallery images as we have shown in section 4.1; and (3) the face recognition rate using TYPE 4 is almost constant over the different numbers of facial expressions. This is because the transformed neutral expression images are quite similar with the gallery images regardless of the number of training subjects.

5.3 Face Recognition Results

Table 2 summarizes the face recognition results with 4 facial expressions. We used three different classification methods: nearest neighbor method (NN), linear discriminant analysis followed by NN (LDA+NN), and generalized discriminant analysis followed by NN (GDA+NN). This table shows that the face recognition result using TYPE 4 shows the best performance, while the face recognition

Table 3. Face recognition results with 7 facial expressions

Transformations	Methods	Happy	Surprise	Anger	Disgust	Sad	Fear	Average
TYPE 1	NN	81.69	27.69	96.77	96.77	91.67	90.0	**74.81**
	LDA+NN	60.56	20.00	83.87	87.09	63.89	67.5	58.02
	GDA+NN	31.81	44.73	42.85	57.14	36.00	38.46	40.74
TYPE 2	NN	47.88	33.84	64.51	48.38	63.8	52.5	**49.27**
	LDA+NN	50.7	40.0	54.83	54.83	52.78	47.5	48.9
	GDA+NN	53.21	35.38	54.83	58.06	38.89	52.5	47.8
TYPE 3	NN	50.7	32.3	67.74	51.61	66.67	47.5	50.0
	LDA+NN	50.7	36.92	61.29	64.51	61.11	47.5	51.09
	GDA+NN	53.52	38.46	61.29	70.96	50.0	50.0	**51.82**
TYPE 4	NN	94.36	84.61	100.0	96.77	91.67	90.0	**91.97**
	LDA+NN	85.91	87.69	90.32	90.32	83.33	90.0	87.59
	GDA+NN	85.91	72.31	83.87	87.09	86.11	72.5	80.65

result using TYPE 2 or TYPE 3 shows also a reasonable performance. Moreover, the use of LDA and GDA improves the face recognition performance greatly.

Table 3 summarizes the face recognition results with 7 facial expressions. This table shows that (1) the face recognition rate using TYPE 2 or TYPE 3 is very poor. This is because the number of training subjects are limited and accordingly the transformed neutral facial expression images are considerably different from the input images; (2) the face recognition rate using TYPE 4 is still over than 90%. It shows the effectiveness of our method; and (3) Although we applied the discriminant methods to improve face recognition performance, the face recognition results of using LDA+NN and GDA+NN did not improve the performance of NN. Rather it degraded the face recognition performance. This is because the representational ability of LDA and GDA bases is also limited due to the limited number of training samples of each subject.

6 Conclusion

In this paper, we proposed the expression-invariant face recognition. To achieve expression-invariance, we first extract facial feature vector from the input image using AAM. Then, we obtain the facial expression state of the input facial feature vector by the facial expression recognizer. Then, we transform the input facial feature vector into its corresponding neutral facial expression vector using the direct or indirect facial expression transformation, and convert the neutral facial expression vector into the neutral facial expression image via the AAM reconstruction. Finally, we perform the expression-invariant face recognition by the distance-based matching techniques. From the experimental results, we note that face recognition rate with proposed expression transformation greatly improve that without transformation. In addition, the face recognition rate using our proposed expression transformation is almost constant over the different numbers of training subjects, while that using bilinear model and ridge-regressive bilinear model decrease as the number of training subject decreases.

References

1. Kanade, T.: Picture processing by computer complex and recognition of human face. PhD thesis, Kyoto University (1973)
2. Liu, Y., Schmidt, K., Cohn, J., Mitra, S.: Facial asymmetry quantificatioin for expression invariant human identification. Computer Vision and Image Understanding **91** (2003) 138–159
3. Elad, A., Kimmel, R.: On bending invariant signatures for surfaces. IEEE Transactions on Pattern Analysis and Machine Intelligence **25** (2001) 1285–1295
4. Wang, H., Ahuja, N.: Facial expression decomposition. In: Proc. of IEEE International Conference on Computer Vision. (2003) 958–964
5. Li, X., Mori, G., Zhang, H.: Expression-invariant face recognition with expression classification. Third Canadian Conference on Computer and Robot Vision (to appear) (2006)
6. Abboud, B., Davoine, F.: Face appearance factorization for expression analysis and synthesis. In: Proc. of Workshop on Image Analysis for Multimedia Interactive Services. (2004)
7. Zhou, C., Lin, X.: Facial expressional image synthesis controlled by emotional parameters. Pattern Recognition Letters **26** (2005) 2611–2627
8. Tenenbaum, J., Freeman, W.: Separating style and content with bilinear models, neural computation. Neural Computation **12** (2000) 1247–1283
9. Lee, H.S., Shin, D., Kim, D.: Ridge regressive bilinear model for robust face recognition. Submitted to International Conference on Ubiquitous Robots and Ambient Intelligence (2006)
10. Sung, J.W., Kim, D.: A real-time facial expression recognition using the STAAM. In: Proc. of International Conference on Pattern Recognition. (2006)
11. Kanade, T., Cohn, J., Tian, Y.: Comprehensive database for facial expression analysis. In: Proc. of International Conference on Automatic Face and Gesture Recognition. (2000) 46–53

A High-Speed Parallel Architecture for Stereo Matching

Sungchan Park and Hong Jeong

Pohang University of Science and Technology
Electronic amd Electrical Engineering
Pohang, Kyungbuk, 790-784, South Korea

Abstract. The stereo matching algorithm based on the belief propagation (BP) has the low matching error as the global method, but has the disadvantage of a long processing time. In addition to a low error of less than 2.6% in the Middlebury image simulation, a new architecture based on BP shows a high-speed parallel VLSI structure of the time complexity O(N), at properly small iterations, so that it can be useful as a chip in the real-time application like robots and navigations.

1 Introduction

The stereo matching is the method which finds the corresponding points in a pair of images to locate the 3D positions. It can be mainly separated into the local and the global matching part [1]. The former is based on the winner-take-all (WTA) method and can be processed in the real-time, but produces big errors [2] [3]. Some VLSI architectures [2] [4] exist in this area.The latter finds the approximated global minimum energy in the full image, see for example, the graph cuts [5] and the belief propagation (BP) [6] approaches. It produces excellent results but takes a long time on a single CPU of. PC. In the real-time applications like robots, the high-speed small compact system is necessary. We will present a new high-speed parallel VLSI architecture for the stereo matching based on BP.

2 Problem Formulation

Markov random field(MRF) energy model of stereo matching can be represented as follows given the left and right image and the parameter C_d, C_v, K_d, K_v [6].

$$\hat{d} = \arg\min_{d} E(d), E(d) = \sum_{p,q \in N} V(d_p, d_q) + \sum_{p \in P} D(d_p), \tag{1}$$

$$D(d_p) = \min(C_d|g^r(d_p) - g^l(p)|, K_d), V(d_p, d_q) = \min(C_v|d_p - d_q|, K_v). \tag{2}$$

$D(d_p)$ is the data cost of the label d_p at the pixel p in the image P , and $V(d_p, d_q)$ is the discontinuity cost between the label d_p and d_q of the neighbour nodes N. The disparity $\hat{d}$ can be estimated using the following BP.

$$m^t_{pq}(d_q) = \min_{d_p} \left(V(d_p, d_q) + D_p(d_p) + \sum_{s \in N(p) \backslash q} \left(m^{t-1}_{sp}(d_p) - \alpha \right) \right), \tag{3}$$

$$\alpha = \sum_{d_p} m^{t-1}_{sp}(d_p), \tag{4}$$

$$\hat{d}_p = \arg\min_{d_p} \left(D_p(d_p) + \sum_{s \in N(p)} m^T_{qp}(d_p) \right), \tag{5}$$

where $N(p)\backslash q$ denotes the neighbours of p other than q, and α is the normalization value. The message $m^t_{pq}(d_q)$ is calculated at iteration t and sent from the node p to the neighbour node q. After T iterations, the $\hat{d}_p$ at each node is decided using Eq. (5).

3 Architecture of Stereo Matching

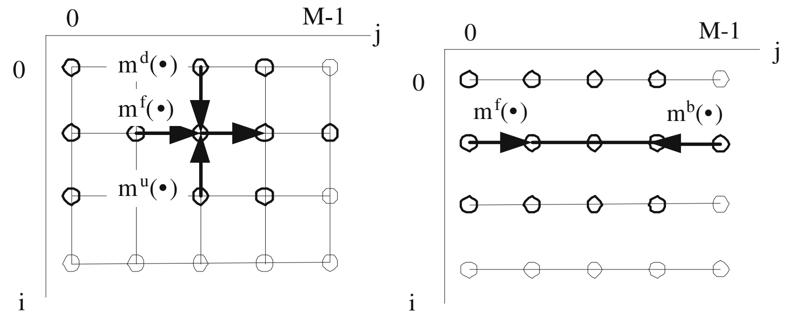
BP can be separated into two methods mainly as the update style [7]. The first way is to update all the nodes synchronously and in parallel, and the second thing is to update sequentially in the inward direction from the leaf to the root and the reverse direction on the tree. The sequential method at each update is not to calculate the messages of all the node so that they can be propagated fully in the small number of operations. In tree case, both algorithms produce exact solutions [7] [8]. Thus there are equivalent to the dynamic programming technique.

In Fig. 1(b) the optimal diparity solution is calculated on the 1D markov chain by the sequential update of the forward and backward directions similar to the dynamic programming(DP) based stereo vision [1]. As this DP based algorithm, the markovian dependence of the horizontal scanline direction is more important than the vertical part in the stereo matching [1]. The vertical smoothness constraint between the scanlines is also necessary but this propagation range doesn't need to be long to just eliminate the streak noises. Applying the hypertree structure to each scanline as shown in Fig. 1(c), the smoothness constraint messages are possible to be propagated through the connection between hypertrees. It has the overall small iteration times in the MRF due to the sequential full propagation inside of the tree and the short vertical propagation outside of it. Another hypertree based algorithm [8] also shows the fast convergence. In this paper we didn't use the reparametized potential functions but reuse the updated messages.

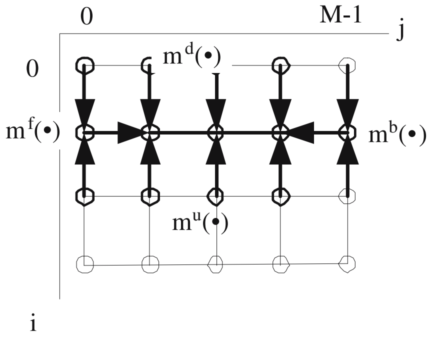
For more explicit expression considering the image coordinate, the message notations like $m^t_{pq}(d_q)$ will be changed as follows. If the node q is changed to (j, i), the state d_q to k,and the iteration step of the vertical message propagation between hypertrees is denoted to f, they can be represented as the forward message $m^f(k, j, i, f)$, the backward message $m^b(k, j, i, f)$, the upward message $m^u(k, j, i, f)$, and the downward message $m^d(k, j, i, f)$ when considering the message propagation direction from the node q to the node p like Fig. 1(c).

Fig. 2 shows the parallel processing of P lines. Using the processor index p, we can change the image coordinate (j, i) to $(j, q * P + p)$ at the step q.

We will present a VLSI parallel pipeline algorithm based on the previous explanations. Let us calculate the message $m^f(k, j)$,$m^b(k, j)$,$m^u(k, j, q * P + p - 1, f)$, and



(a) Synchronous message update at each pixel (b) Sequential message update at each scanline



(c) Our message update of the hypertree structure

Fig. 1. Message update methods

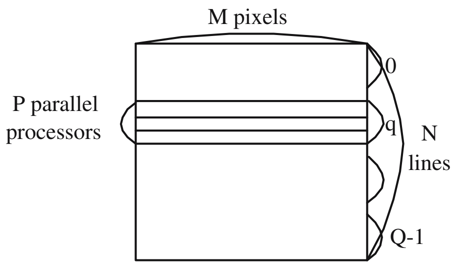


Fig. 2. Parallel processing

$m^d(k, j, q*P+p+1, f)$ at the state k and the node $(j, q*P+p)$ in the N by M image, and find the disparity $\hat{d}$, at the iteration f , the processor p, and the step q.

The messages are updated sequentially inside of the hypertree by the following forward and backward processing and the upward and downward messages are used for the propagation between the neighboring hypertrees.

For $q \in [0, Q-1]$, $f \in [0, F-1]$,

$$D(d) = \min_{d \in [0, d_{max}-1]} \left(C_d \left| g^r(j+d, q*P+p) - g^l(j, q*P+p) \right|, K_d \right)$$
$$V(l,k) = \min \left(C_v \left| l-k \right|, K_v \right).$$

1. Forward processing for state $k = 0, ..., d_{max}-1$ and node $j = 0, ..., M-1$ for the forward message,

$$m^f(k,j) = \min_{d \in [0, d_{max}-1]} V(l,k) + m_{sum}(l),$$
$$m_{sum}(l) = D(l) + m^f(k, j-1)$$
$$+ m^d(k, j-1, q*P+p, f-1) + m^u(k, j-1, q*P+p, f-1).$$

2. Backward processing for state $k = 0, ..., d_{max}-1$ and node $j = M-1, ..., 0$,
   (a) Backward message

$$m^b(k,j) = \min_{d \in [0, d_{max}-1]} V(l,k) + m_{sum}(l),$$
$$m_{sum}(l) = D(l) + m_n^b(l, j+1)$$
$$+ m_n^d(l, j+1, q*P+p, f-1) + m_n^u(l, j+1, q*P+p, f-1).$$

   (b) Downward message

$$m^d(k, j, q*P+p+1, f) = \min_{d \in [0, d_{max}-1]} V(l,k) + m_{sum}(l),$$
$$m_{sum}(l) = D(l) + m_n^f(l,j) + m_n^b(l,j)$$
$$+ m_n^d(l, j, q*P+p, f-1).$$

   (c) Upward message

$$m^u(k, j, q*P+p-1, f) = \min_{d \in [0, d_{max}-1]} V(l,k) + m_{sum}(l),$$
$$m_{sum}(l) = D(l) + m_n^f(l,j) + m_n^b(l,j)$$
$$+ m_n^u(l, j, q*P+p, f-1).$$

   (d) Disparity decision

$$m^b(k,j) = \min_{d \in [0, d_{max}-1]} V(l,k) + m_{sum}(l),$$
$$m_{sum}(l) = D(l) + m_n^f(l,j) + m_n^b(l,j)$$
$$+ m_n^d(l, j, q*P+p, f-1) + m_n^u(l, j, q*P+p, f-1).$$

The parameters which are not used for the memory storage are omitted. For instance $D(d, j, q*P+p)$is replaced with $D(d)$. $m_n(\cdot)$ means $m(\cdot) - \alpha$ as the normalized message, and d_{max} is the total disparity level number. The total step Q is equal to the image line N divided by the processor number P. This algorithm will be explained together with the following architectures.

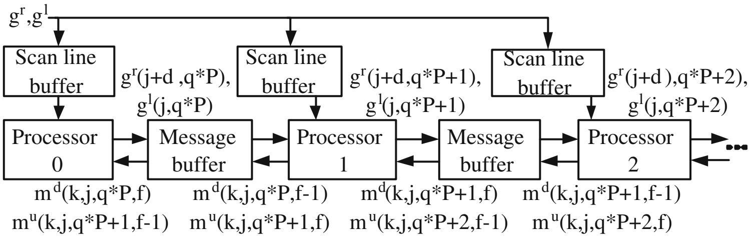


Fig. 3. Parallel architecture of stereo matching

As shown in Fig. 3, the P processors calculate the message m^d,m^u in the parallel, receiving the left and right pixel data from the P scan line buffers, and reading and writing with the P message buffers. The processor consists of processor PE^f, PE^b, PE^u,PE^d, and PE^o. Using the image data and the neighborhood messages, PE^f calculates the message m^f in the forward time and PE^b, PE^u ,PE^d , and PE^o calculate the each message m^b, m^u,m^d and disparity $\hat{d}$ in the backward time.

4 Architecture of Processing Element (PE)

The PE is the basis logic which calculates the messages m^f , m^b , m^u , and m^d at each node. The main processing part at node j for $k = 0, ..., d_{max} - 1$ is as follows, when $V(l, k) = min(C_v|l - k|, K_v)$,

$$m(k, j) = \min_{d \in [0, d_{max}-1]} V(l, k) + m_{sum}(l).$$

By the recursive backward and forward skills of the distance transform [6], the time complexity can be reduced from $O(D^2)$ to $O(5D)$ for D disparity levels. We optimized it to $O(2D)$ with a pipeline structure as follows. That is, we need 2D clocks for calculating the message $m(k, j)$, which is denoted as $m_o(k)$ after.

In the forward initialization, $D_1(-1) = B, D_2(-1) = B$ (B is as big as possible).

For t from 0 to D-1 at the forward processing time,

$$D_1(t) = \min(m_{sum}(t), D_1(t-1) + C_v), \quad (6)$$

$$D_2(t) = \min(m_{sum}(t), D_2(t-1)), \quad (7)$$

$$m_f(t) = D_1(t), \quad (8)$$

$$m_f(-1) = D_2(D-1) + K_v. \quad (9)$$

In the backward initialization, $D_3(-1) = B, D_4(-1) = B$.

For t from 0 to D-1 at the backward processing time,

$$D_3(t) = \min(m_f(D-1-t), D_3(t) + C_v), \quad (10)$$

$$m_o(t) = \min(D_3(t-1), m_f(-1)), \quad (11)$$

$$D_4(t) = m_o(t) + D_4(t-1), \quad (12)$$

$$m_o(-1) = D_4(D-1)/D. \quad (13)$$

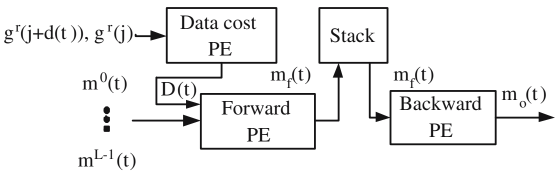
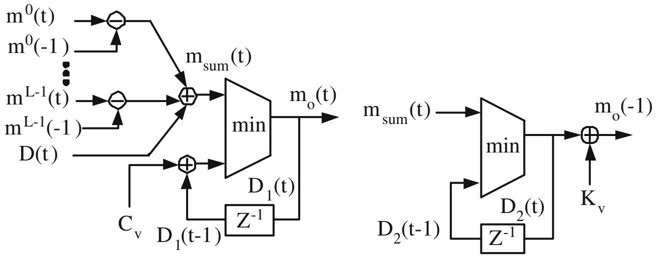
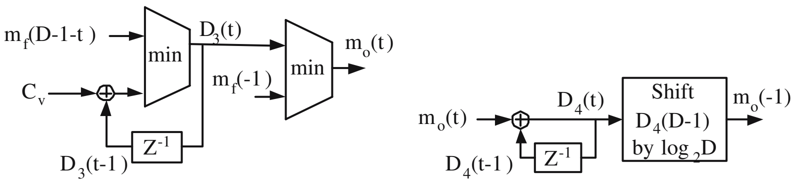


Fig. 4. Internal architecture of PE



(a) Cost processor (b) Parameter calculator

Fig. 5. Forward processor



(a) Cost processor (b) Parameter calculator

Fig. 6. Backward processor

The following VLSI architecture will be described together with this algorithm.

Fig. 4 shows the basis PE architecture of PE^f, PE^b, PE^u, and PE^d. The data cost PE calculates the data cost D(t) from the left and right image pixels. The forward PE reads the messages and the data cost, and outputs the forward cost $m_f(t)$ and saves it to the stack and finally the backward PE reads it from the stack and calculates $m_o(t)$.

In the forward processor architecture, Fig. 5(a) and Eq. (6) show the cost processor and sequences that output the minimum value between $m_{sum}(t)$ and $D_1(t-1)+C_v$ after each message $m^0(t),...,m^{L-1}(t)$ are subtracted by the parameter $m^0(-1),...,m^{L-1}(-1)$ for the normalization and added together with the matching cost $D(t)$ to calculate $m_{sum}(t)$.

In Fig. 5(b) and Eq. (7), the logic calculates the parameter which is used in the backward time. Fig. 6 describes the backward processor's architecture. As shown in Fig. 6(a), Eq. (10), and Eq. (11), the cost processor reads the $m_f(D-1-t)$ from the stack and calculates the minimum value $D_3(t)$ and outputs the minimum value between $D_3(t)$ and the parameter $m_f(-1)$. In Fig. 6(b), Eq. (12), and Eq. (13), after the message $m_o(t)$ is summed up, it is divided by the disparity level D. If the disparity level is 2's exponent, we can replace the divider with the bit shifter. This normalization parameter will be used for the for-ward time afterward. BP [6] has the total time complexity O(NMDT) given N by M image, and D disparity levels and T iterations. Its complexity reduces to O(NMDT/P) by the P processors, so that it can process in the high speed. Furthermore, the real-time processing is possible with O(NM) when P equal to DT is properly small. The iteration number is related to the vertical message propagation. As shown the experiment, the small iterations are enough for the good results.

5 Experimental Results

We tested our system using the following Middlebury bench mark quantitatively and qualitatively. In Fig. 7 and Table 1, the synchronous BP and ours produce the small error difference and the results superior to the WTA [3] in the Tsukuba image. The

Table 1. Error rate of several methods

methods(iter.)	BP(100)	BP(30)	Ours(7)	WTA(1)
error	2.12%	2.6%	2.6%	4.25%

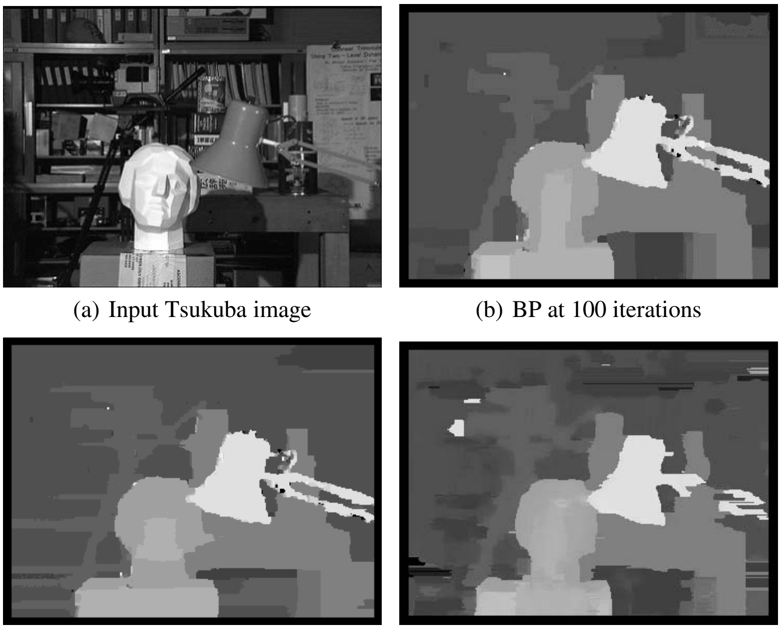

(a) Input Tsukuba image (b) BP at 100 iterations
(c) Our method at 7 iterations (d) WTA

Fig. 7. Several disparity image output

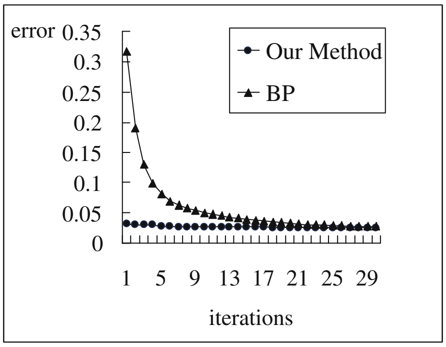


Fig. 8. Comparison of convergence rate

error rate represents the percentage of disparity error of more than 1. When comparing the convergence error between BP and our method in Fig. 8, while BP needs more than 30 iterations, ours is enough for 7 iterations because of the horizontal asynchronous update for the fast convergence. The 112 processors have to be implemented for the real-time processing at 16 disparity levels.

6 Conclusions

The BP is powerful and widely used in the area of the image processing.Although BP produces the good error performances, the VLSI architecture has not been researched yet. In this letter, we explained the parallel VLSI architecture and algorithm for the stereo matching which shows the low matching error and the high-speed performance, and tested it with the several simulation results.

References

1. D. Scharstein and R. Szeliski: A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. International Journal of Computer Vision **47**(1-3) (2002) 7-42.
2. S. Kimura, T. Shinbo, H. Yamaguchi, E. Kawamura, and K. Naka: A Convolver-Based Real-Time Stereo Machine (SAZAN). Proc. Computer Vision and Pattern Recognition **1** (1999) 457-463
3. Hirschmuller, H.: Improvements in real-time correlation-based stereo vision. IEEE Workshop on Stereo and Multi-Baseline Vision. Dec. (2001) 141 - 148
4. Hariyama, M., et al.: Architecture of a stereo matching VLSI processor based on hierarchically parallel memory access. The 2004 47th Midwest Symposium on Cir-cuits and Systems **2** (2004) II245 - II247
5. Kolmogorov, V. and Zabih, R.: Computing visual correspondence with occlusions using graph cuts. In ICCV **2** (2001) 508- 515
6. Felzenszwalb, P.F. and Huttenlocher, D.R.: Efficient belief propagation for early vision. Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition **1** (2004) I261 - I268

7. M.I. Jordan, An Introduction to Probabilistic Graphical Models, in preparation.
8. Martin J. Wainwright, Tommi Jaakkola, Alan S. Willsky: Tree-based reparameterization framework for analysis of sum-product and related algorithms. IEEE Transactions on Information Theory **49**(5) (2003) 1120-1146

Light Simulation in a Distributed Driving Simulator*

Stefan Lietsch, Henning Zabel, Martin Eikermann, Veit Wittenberg, and Jan Berssenbrügge

University of Paderborn
Paderborn Center for Parallel Computing, HNI and C-Lab
slietsch@upb.de, henning@c-lab.de, meiker@upb.de, bsimpson@upb.de, jan@hni.upb.de

Abstract. In this paper we present our work on modularizing and distributing a VR application - the Virtual Night Drive simulator. The main focus in our work is the simulation of headlights. The realistic but still interactive visualization of those lights is essential for a usable driving simulator at night. Modern techniques like pixel and vertex shaders and volume rendering help to realize the complex task of light simulation. But there are still scenarios, especially when having several cars with headlights in one scene, that require distributed setups to run the simulation in an interactive way. In this paper we present an architecture that supports several approaches of distributed light simulation, compare it to existing systems and give an outlook on what is left to do.

1 Introduction

The simulation of real world processes plays a very important role in today's business and science life. It for example helps developers and researchers to build, test and evaluate products much easier and cheaper than before. The computation of these simulations becomes very complex soon and the produced data gets unmanageable. Thus much research on visualization has been done to make the data produced by the simulation easier to understand. By combining High Performance Simulation and High Performance Visualization researchers and developers get powerful systems that significantly ease their work and help to share costs. One special form of simulation is Virtual Reality (VR). VR applications are specialized simulations that allow users to interact in virtual environments in real time. Most of those systems are highly integrated and tightly coupled and designed for a very special purpose. Our approach is to split up this tight coupling and build a more universal and scalable Virtual Reality system. This also includes the possibility to connect different simulations to this system and see how they interact.

As an exemplary Virtual Reality application the Virtual Night Drive (VND) system was chosen. This specialized driving simulator is described in [1] and in

* Partially supported by the Ministry of Innovation, Science, Research and Technology of the State of North Rhine-Westphalia, Germany under the Target Agreement II "Distributed Visualization and Simulation (VisSim)".

[2] and was developed at the Heinz Nixdorf Institute, Paderborn in cooperation with the Hella KGaA Hueck & Co. Its main focus is the realistic simulation of headlights on real-world tracks to do physiological tests and to let engineers evaluate prototypes of new headlights. There are several characteristics of headlights that need to be considered when designing a simulator. It is nearly impossible to visualize the complex light distribution with traditional computer graphics lighting and shading methods like point light sources or Phong shading. Instead a system was developed that takes advantage of programmable pixel and vertex shaders to project the light of a headlight onto the scene per pixel. This provides a very realistic impression for test people and engineers and allows the differentiation of several types of headlights. It is also possible to check the compliance of the headlights to strict standards in a very early stage of development (e.g. dimmed headlights may not beam over a certain horizon).

The first version of the simulator works quite well as long as only one car with headlights is simulated. For more cars the light simulation requires more computational power than a single CPU/GPU workstation can provide. Additionally, it is not trivial to realize effects like blending with the existing shader approach since no real light sources are utilized. But especially for the psychological testing it is extremely important to have more than one car with realistic headlights to simulate effects like oncoming traffic or colones of vehicles which are really important in everyday life. This lead to the idea of distributing the whole system onto several computers and bundle their computational power to provide an interactive system for complex scenarios. By introducing a communication server we created a platform to distribute instances of visualization and simulation. On this basis we developed and partially implemented several methods of distributed light simulations.

In the following we will present the architecture of our system and compare it to existing projects in this area. We give examples on how new simulations can easily be integrated and line out what needs to be considered in the special case of Virtual Reality. Afterwards we focus on the (distributed) simulation of light in the simulator. Finally we give an outlook on what still needs to be done and a conclusion of our work.

2 Related Work

A lot of work on distributed visualization especially on graphic clusters has been done and many interesting techniques were developed. But most of this work focussed either on the visualization of Computational Fluid Dynamics or Finite Element simulations or on volumetric data representation of medical imagery e.g. from Computer Tomography. Starting 1995 when Virtual Reality systems became real enough to be used, a lot of research was done on driving simulators. One of the first was the Iowa Driving Simulator [3] which already offered a modular and flexible architecture. Kuhl et al. proposed a bus system called Iowa Driving Simulator Control program (ICON) [4] that interconnects the different subsystems. These subsystems include all relevant functionalities (Dynamics, Visualization, Control etc.) and databases (Objects, Terrain, Scenario etc.) of the simulator and can be exchanged to test different setups. This architecture provides a scalable and flexible platform for the highly integrated Iowa Driving Simulator. The Iowa Driving Simulator was enhanced by new hardware and software and is now known as the National Advanced Driving Simulator (NADS) or ”the worlds most sophisticated research driving simulator” [5] and is exemplary for a variety of other driving simulators all over the world. However its main focus, as is most of the current driving simulators, is to simulate the physics and the visualization of driving at daytime. Very high effort is put into realistic haptic behavior by building huge Hexapod setups and putting whole cars into big 360 degree visualization domes. This again leads to very specialized systems that are tightly coupled to certain hardware and can only be used at one site.

The main focus of our research lies on on the interactive simulation of light within certain scenarios - realistic dynamic and haptic behavior are needed but of secondary importance. Other applications of our research could therefore be the interactive illumination of the interior of cars or architectural studies with realistic real-time lighting. Thus, we decided to implement a more universal system that acts as a platform for our research on distributed visualization and light simulation.

There are already really good systems for (semi-)automatic distribution of visualization such as Chromium [6] or VR Juggler [7]. Both enable their users to realize complex multi-screen, CAVE [8] or Stereoscopic Display setups (nearly) independent of the Visualization software generating the OpenGL stream. But both systems lack two features that are essential for our scenario:

– Only visualization is considered. To also include simulation and input handling additional, independent systems are needed
– Both systems completely base on distributing the OpenGL stream which is good for transparency but often causes performance issues especially when dealing with big scenes or shader code as in our case.

Therefore we designed and implemented a new platform that satisfies all our needs and that can be adopted to future developments. The architecture of this platform and its features is presented in section 3.

To complete the related work section we refer to research of the Renault Research Center in France. Lecoq et al. present a real-time simulation of automotive headlights that allows various testing scenarios for the companies newly developed headlights ([9] and [10]). However the proposed system bases on traditional lighting methods and needs highly detailed scene models with huge amounts of polygons to achieve acceptable realism. This limits the architecture in performance and scalability. Additionally there is no indication that more than one car with realistic headlights is supported. This is one of the main features our system supports.

3 System Architecture

The initial objective of the Virtual Night Drive application was to give engineers an early impression of new headlight developments. The requirements have evolved since then to more complex scenarios including visualization on stereo displays, tiled screens and cave environments. In addition, support for a variety of input methods such as force feedback wheels (USB), an industrial force feedback steering wheel (CAN-Bus), an existing drive simulator (Smart, L-Lab) and keyboard control had to be integrated. Furthermore a multiuser system was needed to do more extensive psychological tests and to simulate more complex scenarios.

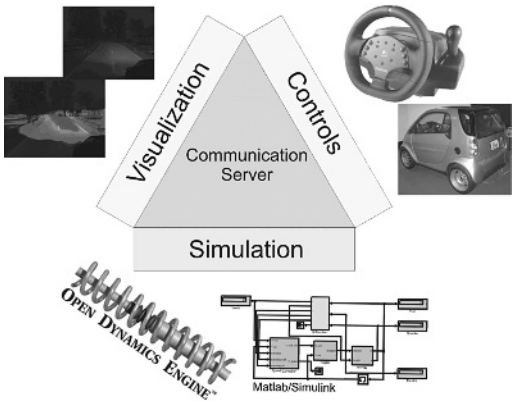


Fig. 1. Overview of the distributed Visualization and Simulation Architecture

In order to achieve the desired flexibility and meet our requirements it was necessary to break down the monolithic design of the Virtual Night Drive application into individual components. To enable a uniform communication between these components we developed the TCP/IP based communication server COMMUVIT [1]. The components simulation, visualization and interaction are separated from each other and executed in different tools as shown in figure 2. COMMUVIT exchanges data between the tools, with the help of a variable map. This universal tool structure is the base of a construction kit for creating interactive visualizations. The network based architecture also allows the distributed execution of this environment, which is an important requirement for the interactive, multiuser simulation. In the following sections we will describe each of the components in detail.

3.1 COMMUVIT

The development of COMMUVIT started within the collaborative research center SFB 614 „Self-optimizing concepts and structures in mechanical engineering"

[1] Communication server for a Simulation and Visualization Toolset.

at the University of Paderborn[2] in the subproject called "Virtual Prototyping". The main goal was to utilize Virtual Reality on the one hand to simplify the design of a mechatronic system and on the other hand to support the analysis of simulations. Especially the second goal required a generic interface between visualization (VR) and simulation. The interface of COMMUVIT supports the runtime platform for simulation and execution IPANEMA that is used within this research center. To provide a broader base for different applications COMMUVIT also supports the connection to the standard simulation systems Matlab/Simulink, PTOLEMY II and SystemC. COMMUVIT has been successfully used for interactive analysis of different case studies (e.g. [11]). The COMMUVIT server offers a universal interface for all components (tools). The protocol used for communication allows controlling calculations within the tools as well as reading output values and writing new parameters to them. To connect tools to COMMUVIT a so called moderation interface is needed. This can be implemented easily with the help of a C++ or Java library that delivers the above mentioned functionality.

The main task of COMMUVIT is to exchange data between the different tools and to force the tools to follow a global clock. Therefore, COMMUVIT is divided into three parts: (1) a master thread that controls time and coordinates communication, (2) a variable map: this map stores the current values of variables during data transfer between the tools and also defines the mapping between input and output variables, (3) a communication process for each master port: these processes executes the communication with the connected tools.

The master process separates the time in intervals of fixed size Δt. For given time t it initiates the sequential execution of the following steps:

1. Write current values from the map as parameters to the tool
2. Setup the time interval $[t, t + \Delta t]$ for calculation and initiate the calculation
3. Wait until the tool has finished calculation
4. Read the output variables and store them back in the map

Each step is executed synchronously, that means the master process initiates one single step and all communication processes execute the necessary data transfers with its corresponding tool. If all communication is done the master continues with the next step. After one sequence is executed the global time is increased to $t = t + \Delta t$ and the execution restarts with the new time interval.

3.2 Simulation

To enable a realistic driving simulator a dynamic simulation was integrated into the Virtual Night Drive. This system simulates effects like suspension and a lifelike drivability from a physical model. In the initial version of the Virtual Night Drive this simulation was realized by proprietary software called Vortex. We have adapted this version to our COMMUVIT environment and separated

[2] The SFB614 is founded by the ,,Deutsche Forschungsgemeinschaft" (DFG - German Research Foundation).

the Vortex simulation from the visualization part. This allows us to replace the simulation by a more accurate one (e.g. Matlab/Simulink) or an open source system like Open Dynamics Engine without changing the visualization and interaction parts anymore. Moreover, since not only one but many cars need to be simulated, possibilities to distribute the dynamics simulation are given by the architecture of our environment. The distribution of a single dynamic simulation itself is another considerable option

3.3 Input Devices

Simple mouse/keyboard input is not sufficient for a realistic driving simulator. Therefore, a support for special input devices like steering wheels and pedals is needed. Depending on the setup of the environment many different input devices connected to various computers can be used to control different cars. Besides common USB steering wheels, we integrated a real Smart car which is an integral part of a drive simulator in the L-LAB[3] facility of the University of Paderborn. Because the input handling is designed as an independent component we can realize various setups. It is for example possible to let remote users control cars through a network connection. Furthermore, new input devices like an industrial force feedback wheel can be integrated easily.

3.4 Visualization

Though simulation of car dynamics is an important part of the sample application the focus is mainly on the realistic simulation of headlights. Virtual Night Drive bases on OpenSceneGraph to render a test scenario consisting of a virtually reconstructed track and one or more cars. Its modularized architecture allows comfortable addition of new objects, like pedestrians or wild life to the scene. We included all functionalities of the visualization into a component named VND-Vis. Each car in the visualization is linked to a corresponding simulation through COMMUVIT. That means every VND-Vis entity has an interface through which it receives positions, directions, speeds and other parameters of all dynamic objects. The visualization component only displays these objects at the right place. COMMUVIT also allows transmitting simulation data to multiple, distributed VND-Vis instances.

3.5 Distributed Visualization

High-resolution visualization on Tiled Walls can be realized really easy with the proposed architecture. For each screen an instance of the VND-Vis component is launched. Each instance gets the information which tile of the whole display it has to render at initialization time. This group of instances acts like a single visualization to the corresponding simulation. That is, all instances get the same data from the simulation including speed, direction and position of the car

[3] L-Lab is a public private partnership between University of Paderborn and Hella KGaA Hueck & Co, Germany - a manufacturer of automotive headlights.

as well as state parameters such as day/night, headlight type etc. Thereby we achieve best possible scalability for this scenario, since we can nearly keep up the frame rates of a non-distributed visualization in our distributed architecture independent of the number of tiles. In comparison to frameworks like Chromium this is a great benefit that results from reduced amount of data that has to be transferred. While Chromium needs to transfer the OpenGL-Stream which can be up to 1 Gbit/s (without optimizations like display lists etc.) per tile our architecture gets by with only small amounts of data containing the positions of the objects. In our test scenario which is 6 nodes (Dual Opteron, Quadro FX4500) driving 6 screens (1280x1024) we got frame rates around 150 Hz with our architecture vs. 20-30 Hz with Chromium. This isn't really a fair comparison since Chromium tries to distribute non-parallel applications but it shows that its really worth to put effort in distributing Virtual Reality applications in the way we do.

For stereoscopic display wall or CAVE setups our architecture can provide similar functionalities as for the tiled wall described above. The difference is that in addition to the viewport we also have to specify different camera positions and directions at initialization time. This allows us to realize really complex settings without having to rewrite source code and without seriously loosing performance.

4 Light Simulation

The automotive industry uses VR-technology for driving simulation within the development and evaluation of new car concepts. Mostly these systems visualize a car driving at daylight. A visualization of nightly scenes is often done by using conventional, OpenGL-based lighting effects. But standard OpenGL light sources only provide a simple light model, i.e. a conic emission characteristic, which cannot emulate a headlight's complex luminance distribution. Additionally they lack of distance-dependent attenuation for the luminance and complex luminance distribution models, which are essential to realistically simulate headlights.

A further problem is a permanently changing position and orientation of the light source and the user's viewpoint within a driving situation at night. Thus, it is not sufficient to calculate the illumination of the whole scene in advance, because the illumination conditions change drastically depending on the headlight's position and orientation.

4.1 Shader-Based Light Simulation

To solve these problems, we developed the following approach shown in figure 6. The basic idea is to use a homogenous emitting OpenGL light source as a slide projector. The luminance distribution functions as a slide being projected onto the landscape in the nocturnal scenery. The slide filters the homogenous OpenGL light source to provide the complex lighting characteristics. To add the headlights illumination onto the scenery the standard rendering pipeline is

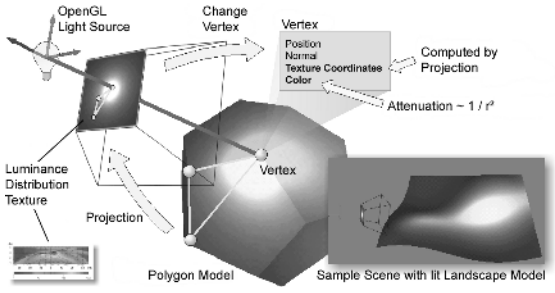


Fig. 2. Overview of the distributed Visualization and Simulation Architecture

extended by additional steps after an object has been transformed, textured, and ambiently lighted. The steps contain roughly the following operations:

1. Project the vertices of a scene object by an inverse perspective transformation onto its corresponding point on the luminance distribution texture.
2. Determine the illumination intensity according to the point's position on the luminance distribution texture.
3. Calculate the final illumination of the scene objects by using a vertex shader. The vertex shader considers the illumination intensity value from step 2, the color of the corresponding ambient lighted object vertex, and the distance between the vertex and the observer for the distance-dependent attenuation of light.

4.2 Distributed Shader Visualization

One of the biggest limitations we face in the Virtual Night Drive application are the shader programs used for the simulation of the headlights. We use textures of up to 2048x2048 pixels which represent the light distribution of one headlight. This high resolution is needed to provide a realistic visualization of the headlights and to study the differences between different types of headlights. Each car has two headlights that is two textures and having a scene with more than one car results in significant frame rate drops. Our approach is to use more than one GPU to calculate the distribution of light for multiple cars in the scene. Starting off with one car per GPU in later versions we may be able to optimize the code to handle two or three cars.

The distribution takes place as follows:

1. For each simulated car one instance of VND-Vis is started.
2. All instance have the same view, that is viewport, frustum, camera position, of the scene.
3. Each VND-Vis has the appropriate light distribution textures for one car.
4. Each instance renders a grey-scale image of its light distribution, not including scene textures or colors.
5. All rendered images are sent to the additional VND-Vis instance where they are combined and mapped to the unlighted scene.

One difficulty of this method is the reassembly of the final frame because we have to transfer and combine n high resolution pictures depending on how many nodes are used. This implicates the need of very fast interconnects between the visualization nodes and optimized algorithms to combine the rendered images. In our case the visualization nodes are connected through a high speed InfiniBand network that will be sufficient in terms of bandwidth and latency. For the second problem, the combination of the frames, we are currently implementing a software that can do this task with the help of GPUs or FPGAs. The goal is to be able to distribute the shader computation and thereby breach the shader limitation without the user noticing it.

4.3 Volumetric Light and Fog

To further improve realism in the simulation of luminance, one must consider the three-dimensionality of light. Especially for headlights of cars the light cones have very specific shapes that have effects on the illumination of the landscape, blending of the oncoming traffic or refractions in foggy or rainy situations. In order to simulate three-dimensional light we consider using specialized volume rendering methods. Therefore, the relevant space in front of the car is partitioned into a sufficient amount of voxels that all have a light value. These light values are computed from the given light distribution texture by shooting attenuating rays through the texture where they get their initial light values from. Each voxel stores the light value of the ray at its current position. Thereby we get a highly realistic three dimensional-luminance distribution. By varying the amount of rays or the size of the voxels we can adjust the simulation in terms of performance and quality. This computation only need to be done when the light distribution texture is changed. Once calculated the 3d light distribution is projected into the scene by shading technologies. To simulate fog or rain, particle systems can be employed where each particle gets transparency and lighting information from the voxel at its current position. This approach is still at an early stage of development and will be published in more details soon.

5 Conclusion and Outlook

In this paper we presented an approach to use the power of distributed computing and visualization for an exemplary driving simulator at night. We designed a component based architecture that supports us in researching new forms of distributed visualization and the simulation of light. This architecture can easily be reused in similar scenarios and applications because of its modular design and its various interfaces. By distributing the visualization component we are able to on the one hand visualize the VR application on various high-end devices (tiled walls, stereoscopic displays, caves) without having to change the source code and on the other hand can breach limitations that current hardware has.

This for example helps to do interactive light simulations, as described above, that were not possible on non-distributed systems before. On the basis of this architecture we will develop other techniques for luminance simulation such as the volumetric light approach also briefly described in this paper.

References

1. Berssenbruegge, J.: Virtual Night Drive - A Method for Displaying the Complex Light Characteristics of Modern Headlight Systems Within a Simulated Night Drive. PhD thesis, Universitiy of Paderborn, Faculty of Mechanical Engineering (2005)
2. Gausemeier, J., Berssenbruegge, J., Matysczok, C., Poehland, K.: Real-time representation of complex lighting data in a night drive simulation. In: Proceedings of the workshop on Virtual environments 2003, ACM Press (2003) 65 – 70
3. Kuhl, J., Evans, D., Papelis, Y., Romano, R., Watson, G.S.: The iowa driving simulator: an immersive research environment. Computer **28** (1995) 35–41
4. Kuhl, J., Papelis, Y.: A real-time software architecture for an operatorin-the-loop simulator. In: Proceedings of Workshop on Parallel and Distributed Real-Time Systems (in conjunction with 1993 International Parallel Processing Symposium). (1993) 117–126
5. NADS: National advanced driving simulator - the most sophisticated driving simulator in the world (brochure). Website University of Iowa (http://www.nads-sc.uiowa.edu/) (2006)
6. Humphreys, G., Houston, M., Ng, R., Frank, R., Ahern, S., Kirchner, P.D., Klosowski, J.T.: Chromium: A stream-processing framework for interactive rendering on clusters. ACM Transactions on Graphics **21** (2002) 693–702
7. Bierbaum, A., Just, C., Hartling, P., Meinert, K., Baker, A., Cruz-Neira, C.: Vr juggler: a virtual platform for virtual reality applicationdevelopment. In: Proceedings of IEEE Virtual Reality, 2001, IEEE (2001) 89–96
8. Cruz-Neira, C., Sandin, D.J., DeFanti, T.A., Kenyon, R.V., Hart, J.C.: The cave: audio visual experience automatic virtual environment. Commun. ACM **35** (1992) 64–72
9. Lecocq, P., Kelada, J.M., Kemeny, A.: Interactive headlight simulation. In: Proceedings of Driving Simulation Conference, 1999. (1999) 173–180
10. Canry, M., Cherfan, S., Lecocq, P., Kelada, J.M., Kemeny, A., Dubrovin, A., Lelev, J., Prevost, A.: Application of real-time lighting simulation for intelligent front-lighting studies. In: Proceedings of Driving Simulation Conference, 2000. (2000) 333–343
11. Bauch, J., Radkowski, R., Zabel, H.: An explorative approach to the virtual prototyping of self-optmizing mechatronic systems. In: Proceedings of ProSTEP iViP Science Days 2005 - Cross Domain Engineering, Darmstadt, 2005. (2005)

Self-adaptive RBF Neural Networks for Face Recognition

S. Gharai[1], S. Thakur[2], S. Lahiri[3], J.K. Sing[1,*], D.K. Basu[1], M. Nasipuri[1], and M. Kundu[1]

[1] Dept. of Computer Science & Engineering, Jadavpur University, Kolkata, India
[2] Netaji Subhas Engineering College, Kolkata, India
[3] TCS SEEPZ, Andheri West, Mumbai, India

Abstract. A self-adaptive radial basis function neural network (RBFNN)-based recognition of human faces has been proposed in this paper. Conventionally, all the hidden layer neurons of an RBFNN are considered to generate outputs at the output layer. In this work, a confidence measure has been imposed to select a subset of the hidden layer neurons to generate outputs at the output layer, thereby making the RBFNN as self-adaptive for choosing hidden layer neurons to be considered while generating outputs at the output layer. This process also reduces the computation time at the output layer of the RBFNN by neglecting the ineffective RBFs. The performance of the proposed method has been evaluated on the ORL and the UMIST face databases. The experimental results indicate that the proposed method can achieve excellent recognition rates and outperform some of the traditional face recognition approaches.

1 Introduction

Face recognition is a process, by which the identification of a person is determined from the faces stored in a large database. It has many potential applications, such as, surveillance, credit cards, passport, security, etc. Many approaches for face recognition have proposed since last two decades [1]. Neural networks have also been used successfully for face recognition problem [2]-[4]. The advantage of using the neural networks for face recognition is that the networks can be trained to capture more knowledge about the variation of face patterns and thereby achieving good generalization. In recent times many researchers have used RBF networks for face recognition for its faster learning ability and best approximation property [2]-[4]. However, many of their success rates are not so promising under the variation of pose, orientation, scale and light [3]. This may be due to the fact that the selection of the centers of the hidden layer neurons might not have been done by capturing the knowledge about the distribution of training patterns and variations of face pose, orientation and lighting. Er *et al.* [2] have used principal component analysis (PCA) method with radial basis

[*] Corresponding author: jksing@ieee.org. The author is now at the ECE Dept., University of Iowa, IA, USA as a BOYSCAST Fellow (Ref. No. SR/BY/E-23/05), Dept. of Science & Technology, Govt. of India.

function (RBF) networks for face recognition. The PCA techniques involve some face feature extraction process, which is computationally expensive. The PCA technique also retains unwanted variations due to lighting, facial expression, and other factors [2]. In our earlier work [3], we have used a modified k-means clustering algorithm using point symmetry distance as similarity measure to model the hidden layer neurons of an RBFNN for face recognition. Yang *et al.* [4] have also used RBF neural network for face recognition.

In this work, a self-adaptive radial basis function neural network-based recognition of human faces has been proposed. Conventionally, all the hidden layer neurons of an RBFNN are considered to generate outputs at the output layer. To make the RBFNN self-adaptive for choosing a subset of the hidden layer neurons to be considered while generating outputs at the output layer, a confidence measure has been imposed on the outputs of the hidden layer neurons. Thereby, the computation time at the output layer of the RBFNN gets reduced.

2 Design of the Self-adaptive RBFNN

The RBFNN is a three-layered feed forward neural network. The function of an RBFNN can be viewed as a process, which maps each of the p-dimensional input patterns from input space to a decision space of m-dimension. In doing so, a non-linear function and a linear function are used in the hidden layer and output layer, respectively. Conventionally, a Gaussian function is used as a non-linear function, which is defined as follows:

$$\varphi_j(x_i) = \exp\left(-\frac{\| x_i - c_j \|^2}{2\sigma_j^2}\right), j = 1,2,...,n; i = 1,2,...,N \tag{1}$$

where $\mathbf{c_j}$ and σ_j are the center and the width of the receptive field of the jth neuron of the hidden layer, respectively, n and N are the total number of hidden layer neurons and total number of input patterns, respectively. The output of the kth output layer neuron is generated by the following linear function:

$$z_{ik} = \sum_{j=1}^{n} \varphi_j(x_i) w_{kj} + b_k w_k,\ k = 1, 2, \ldots, m \tag{2}$$

where w_{kj} is the weight of the link between the jth neuron of the hidden layer and the kth neuron of the output layer, b_k and w_k are unit positive bias and weight of the link to the kth output neuron from the bias neuron, respectively.

In this work, we have tried to implement the human cognition process, which, when new information comes, recalls only a few experiences those are highly related to the new information from its vast experiences. To realize the human cognition process, a confidence measure ε has been imposed to select the appropriate hidden layer neurons to be used to generate the outputs at the output layer of the proposed RBFNN. The jth hidden layer neuron is selected for the computation of outputs at the output layer for an input pattern $\mathbf{x_i}$ if $\varphi_j(x_i) \geq \varepsilon$, otherwise it is neglected. Therefore, this approach selects those nonlinear models, which belong to the close neighborhood of the input pattern. This process reduces the number of false positives or intruders; since the less important information (here irrelevant faces) are excluded while computing outputs at the output layer of the RBF networks. Since a subset of the total hidden layer neurons are considered, the computation time at the output layer also gets reduced. The value of ε should be chosen by considering the width of the receptive fields of the Gaussian function used in hidden layer neurons, performance needed, etc. Therefore, considering the above model, the equation (2) becomes as follows:

$$z_{ik} = \sum_{j=1}^{n} \{\varphi_j(\mathbf{x_i})w_{kj} + b_k w_k \,|\, \varphi_j(\mathbf{x_i}) \geq \varepsilon\} \tag{3}$$

It has been seen that the difference between two face images, taken varying pose, orientation, light, etc., of a person is much more than that of two different persons. Therefore, to cover the wide variation in the input space, each of the training images has been chosen as a candidate for a cluster center. A subset of the hidden layer neurons is selected, according to the outputs of their Gaussian functions using equation (3), for each of the training patterns after completion of the very first epoch, while training the network. Since the centers of these Gaussian functions (i.e. the cluster centers) are fixed, the selected subset of the hidden layer neurons, corresponding to a training pattern, remains unchanged during the subsequent epochs during training period. Thus the training process remains stable in successive epochs.

The Euclidean distance between the center of the j^th hidden layer neuron and the center of the other nearest neuron, multiplied by a constant *B*, has been taken as the width of the receptive field of the j^th hidden layer neuron and is defined as follows:

$$\sigma_j = B \text{ x } \min \left\| c_j - c_i \right\| \text{, i, j= 1, 2, …, } n \text{ and i != j} \tag{4}$$

where *B* (>=1) is a constant and its value can be obtained experimentally for which maximum average recognition rate is achieved.

A faster version of the Least-Mean-Square (LMS) algorithm has been used to estimate the weights of the links between the hidden layer and output layer of the proposed self-adaptive RBFNN.

3 Experimental Results

The performance of the proposed method has been evaluated on two popular face databases, the AT&T Laboratories Cambridge database (formerly called ORL face database) [5] and the UMIST face database [10]. All the face images of the two databases have been normalized by sub sampling with a resolution of 16x16 and 256 gray levels [3], [4]. Each face image has been converted into a 1-dimentional array by concatenating rows, where each element of the array represents the gray value of the corresponding pixel. Thus, each face image is represented by a vector of 16x16 = 256 features. The recognition rate (R_{avg}) has been defined as the ratio of the total number of correct recognition by the method to the total number of images in the test set for a single experimental run.

3.1 Performance Evaluation on the ORL Database

The ORL database contains 400 grayscale images of 40 persons. Each person has 10 images, each having a resolution of 92 x 112 and 256 gray levels. Five (s=5) images from each individual of the database are selected randomly for training set and the rest of the face images are included in the test set. Therefore, a total of 200 faces are used to train and another 200 faces are used to test the self-adaptive RBFNN. It should be noted that there is no overlap between the training and test images. In this way five different training and test sets have been generated. Next, the training and test images are exchanged and experiments were repeated once more.

In our first experiment, the value of the threshold (ε) on the output of the hidden layer neurons has been determined for which maximum average recognition rate is achieved. When value of the threshold is set to zero, all the 200 hidden layer neurons are considered for computation of outputs at the output layer. When this value is increased slowly, some of the hidden layer neurons are neglected for computation at the output layer. Fig. 1 shows the average recognition rates of the proposed method for different values of the threshold. The highest average recognition rate (96.25%) has been obtained and average 188 hidden layer neurons are selected for computation of outputs at the output layer when value of the threshold (ε) is set to 0.145.

In the next experiment, we have fixed the widths of the receptive fields. The performance of the proposed self-adaptive RBFNN also heavily depends on the widths of the receptive fields of the Gaussian functions. The larger width means that the RBF is spread out more to cover a larger number of clusters in the input space, resulting in more misclassifications. Similarly, a shorter width also leads to more misclassifications since it will cover a small number of clusters. Fig. 2 shows the average recognition rate of the method by varying multiplicative factor B. It shows that the highest average recognition rate (97.10%) is achieved when B = 1.01.

Due to the characteristics of the face structure, many features may be redundant in the 256-dimensional feature vector. In the third experiment, the performance of the self-adaptive RBFNN has been tested by reducing the feature vector length. We have taken average gray level of 2, 4, and 8 consecutive pixels on each row of the face image resulting in feature vectors of length 128, 64 and 32, respectively. The Average recognition rates and hidden layer neurons are found to be 96.30%, 95.00%, and 91.25% and 182, 172, and 163 for 128, 64, and 32 features, respectively.

The performance of the proposed method has also been evaluated by averaging gray levels of 2, 4, and 8 consecutive pixels on each column of the face image. The Average recognition rates and hidden layer neurons are found to be 96.90%, 87.85%, and 80.15% and 181, 164, and 135 for 128, 64, and 32 features, respectively.

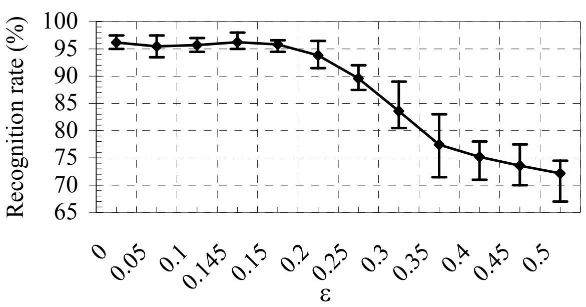


Fig. 1. Recognition rate versus threshold value ε. The upper and lower extrema of the error bars represents the maximum and minimum values, respectively.

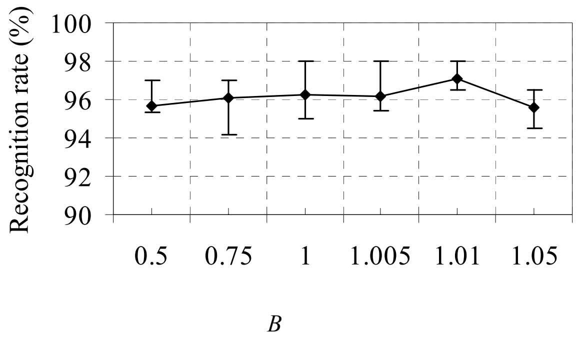


Fig. 2. Recognition rate versus the value of *B*. The upper and lower extrema of the error bars represents the maximum and minimum values, respectively.

The performance of the proposed method has also further been evaluated by averaging gray levels of 2, 4, and 8 consecutive pixels on each row and column simultaneously. The average recognition rates obtained and hidden layer neurons selected are found to be 94.40%, 88.90%, and 53.85% and 174, 160, and 149 for 64, 16, and 4 features, respectively.

In all the above experiments, five (s=5) images per individual have been included in the training set and rests are included in the test set. However, in this experiment we have evaluated the average recognition rates of the proposed method by considering one, two, three, four, six and seven (s=1, 2, 3, 4, 6 and 7) images per individual into each of the training set and rests of the faces into the corresponding test set. For each value of s, 10 different training and test sets have been generated by selecting training images randomly from the database. It should be again noted that there is no overlap between the training images and the corresponding test images. Table 1 shows the average recognition rates of the proposed method and hidden layer neurons selected ($\mathrm{HLN}_{\mathrm{avg}}$), in 10 experimental runs using 256 features for different values of s. With s=6 we have achieved highest average recognition rate of 98.73%.

Table 1. Average recognition rates of the self-adaptive RBFNN for different number of training samples

s	7	6	5	4	3	2	1
HLN_{avg}	269	229	188	145	101	51	39
R_{avg} (%)	98.58	98.73	97.10	96.33	93.96	84.81	75.11

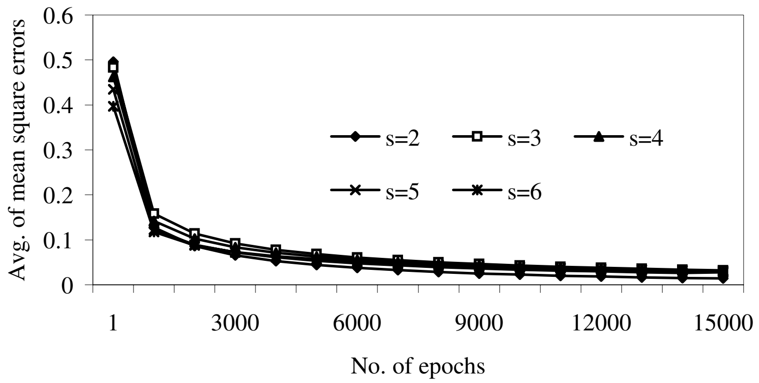


Fig. 3. Average mean square errors (MSEs) of the proposed RBFNN in successive epochs

The training process of the self-adaptive RBFNN remains stable during training period, as discussed in Section 2. The convergence of the proposed method can be visualized by considering the convergence of means square errors (MSEs) at the output layer over the training patterns. For the simplicity, Fig. 3 shows the convergences of average MSE in 10 experimental runs using each of the five different configurations (s=2, 3, 4, 5, and 6) of the method.

Recently, many researchers have used the ORL face database to evaluate the performances of their algorithms [2], [3], [6]-[9]. The performances (in terms of average error rate, E_{avg}) of the proposed method have been compared with the methods reported in [2], [3], [6]-[9], as shown in Table 2. The best values for the CNN [8], NFL [9], M-PCA [7], and PCA+FLD+RBF [2] are based on three, four, ten, and six runs of experiments, respectively. In our earlier work [3], we have achieved some good recognition results in comparison to the others. It should be noted that the results presented in Table 2 are obtained using 200 training and 200 test face images. The average error rate (E_{avg} = 2.90 in 10 experimental runs) of our method is comparable to the other reported methods.

3.2 Performance Evaluation on the UMIST Database

The UMIST face database consists of 575 gray-scale pre-cropped images of 20 people, each covering a wide range of poses from profile to frontal views. Each subject

Table 2. Comparison of error rates with other methods

Approach	No. of simulations	E_{avg} (%)
CNN [8]	3	3.83
NFL [9]	4	3.13
M-PCA [7]	10	2.40
PCA+FLD+RBF [2]	6	1.92
RBF+Point Symmetry [3]	10	2.80
RCGI [6]	1	8.50
Proposed Method	10	2.90
Proposed Method	6	2.50
Proposed Method	4	2.25
Proposed Method	3	2.00

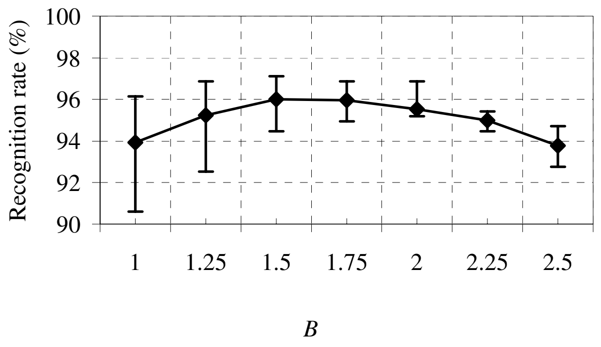


Fig. 4. Recognition rate versus the value of *B*. The upper and lower extrema of the error bars represents the maximum and minimum values, respectively.

also covers a range of race, sex and appearance. Each image has a resolution of 92 x 112 and 256 gray levels.

Eight images per subject have been selected randomly to form a training set of 160 images. Remaining 415 images have been used as the corresponding test set. In this way five different training and test sets have been generated to test the performance of the proposed method. The recognition rate of the proposed method has been evaluated by averaging the recognition rates obtained over these five experimental runs. It should be again noted that there is no overlap between the training and test sets in a particular experimental run. In this experiment, first, widths of the receptive fields of the Gaussian functions have been identified and then value of the threshold of the output of the hidden layer neurons has been fixed. The average recognition rates of the proposed method over the UMIST database by varying the multiplicative factor *B* have shown in Fig. 4. The highest average recognition rate (96%) has been obtained by using $B = 1.50$.

In this section, value of the threshold on the outputs of the hidden layer neurons has been set for which maximum average recognition rate is achieved. Fig. 5 shows the average recognition rates of the method by varying value of the threshold ε. In all experimental runs, the value of B has been set to 1.50. The highest average recognition rate (96.24%) has been obtained and average 121 hidden layer neurons are selected for computation of outputs at the output layer when $\varepsilon = 0.025$.

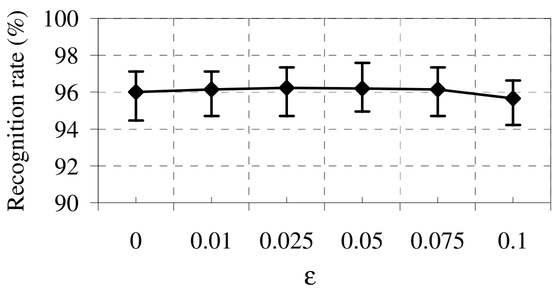


Fig. 5. Recognition rate versus threshold value ε. The upper and lower extrema of the error bars represents the maximum and minimum values, respectively.

The recognition rates of the proposed method have also been tested by reducing feature vector length in row and column-wise. The values of B and ε have been set to 1.50 and 0.025, respectively. The feature vector has been reduced in the same way as it has been discussed in sub-section 3.1. Average recognition rates and hidden layer neurons are found to be 95.71%, 96.48%, and 96.00% and 105, 83, and 60 for 128, 64, and 32 features, respectively for row-wise reduction. Whereas the average recognition rates and hidden layer neurons are obtained as 95.61%, 94.75%, and 93.01% and 112, 103, and 90 for 128, 64, and 32 features, respectively for column-wise reduction.

In this experiment, the average recognition rates of the proposed method have been evaluated by considering four and six (s=4 and 6) images per individual into each of the training set and rest of the faces into the corresponding test set. For each value of s, five different training and test sets have been generated, by selecting training images randomly, from the database. The feature vectors for all the images of the training and test sets have been reduced by taking average gray level of 4 consecutive pixels on each row of the face image, resulting its length of 64 features. The average recognition rates of the proposed method and selected hidden layer neurons, in 5 experimental runs for s = 6 and 4 are obtained as 91.69%, 88.04% and 75, 63, respectively.

A number of methods have also used the UMIST database to evaluate their performances. Table 3 shows the comparison between the proposed method and the methods as reported in [11]. It should be noted that methods presented in [11] have not considered all the 575 images of the database; rather they have selected 25 images from each subject to construct 500 input images. In the case of six subjects, which have less than 25 images, they have generated a few "mirror" images to make the image number in the subject up to 25.

Table 3. Comparison of error rates (%) with other methods

No. of sample(s) / individual	4	6	8
PCA [11]	16.94	13.25	6.40
PCA+FLD [11]	14.24	9.33	5.43
DLDA [11]	11.62	8.99	3.79
2DPCA [11]	10.39	6.86	3.07
2DFLD [11]	7.11	5.46	1.76
Proposed Method	11.96	8.31	3.52

The face recognition method has been implemented in C programming language on a Gateway Intel Pentium III 450 MHz computer with 256 MB SDRAM running on Fedora Core Linux. The method (with 64 features, 121 average hidden layer neurons, 20 output layer neurons, $B = 1.50$ and $\varepsilon = 0.025$) takes approximately 12 minutes to complete a run of 10000 epochs, each one having 160 training images for the determination of the required parameters. On the other hand, it takes approximately 13 minutes to complete the above run when ε is set to zero. Therefore, in the proposed method, the training time of the RBF network is approximately reduced by 7%. Once the parameters are determined, it takes about 2.4 milliseconds to recognize a face. Thus, the present method will be able to recognize faces in interframe periods of video and also in other real-time applications.

4 Conclusion

A face recognition system using a self-adaptive RBF neural network has been presented. Since the difference between the two face images of any person varies widely due to the variations of poses, orientations, expressions, etc., each of the training images has been considered as a candidate for an individual cluster. A criterion has been imposed for selection of a subset of the hidden layer neurons, which is to be considered for computation of outputs at the output layer of the proposed self-adaptive RBFNN. In this way, the number of false positives or intruders and the total computation time at the output layer of the proposed RBFNN gets reduced. The experimental results obtained on the ORL and UMIST face databases with different configurations of the method, show some promising recognition rates in comparison to some other reported methods.

Acknowledgments

This work was partially supported by the CMATER and SRUVM projects of the CSE Dept., Jadavpur University, Kolkata and the BOYSCAST Fellowship 2005-2006, Dept. of Science & Technology, Govt. of India, (Ref. No. SR/BY/E-23/05). The authors would also like to thank Dr. P. K. Saha, ECE Dept., University of Iowa, IA, USA, for his advice and providing necessary computing facilities.

References

1. Samal, A., Iyengar, P.: Automatic recognition and analysis of human faces and facial expressions: A survey, Pattern Recognition, vol. 25, (1992), 65-77.
2. Er, M. J., Wu, S., Lu, J., Toh, H. L.: Face recognition with radial basis function (RBF) neural networks, IEEE Trans. Neural Networks, vol. 13, (2002), 697-710.
3. Sing, J. K., Basu, D. K., Nasipuri, M., Kundu, M.: Face recognition using point symmetry distance-based RBF network, Applied Soft Computing, Elsevier, in Press, 2005.
4. Yang, F., Paindovoine, M.: Implementation of an RBF neural network on embedded systems: real-time face tracking and identity verification, IEEE Trans. Neural Networks, vol. 14, (2003), 1162-1175.
5. ORL face database. AT&T Laboratories, Cambridge, U. K. [Online] Available: http://www.uk.research.att.com/facedatabase.html.
6. Ayinde, O., Yang, Y.-H.: Face recognition approach based on rank correlation of gabor-filtered images, Pattern Recognition, vol. 35, (2002), 1275-1289.
7. Brennan , V., Principe, J.: Face classification using a multiresolution principal component analysis, Proc. IEEE Workshop Neural Networks Signal Processing, (1998), 506-515.
8. Lawrence, S., Giles, C. L., Tsoi, A. C., Back, A. D.: Face recognition: A convolutional neural-network approach, IEEE Trans. Neural Networks, vol. 8, (1997), 98-113.
9. Li, S. Z., Lu, J.: Face recognition using the nearest feature line method, IEEE Trans. Neural Networks, vol. 10, (1999), 439-443.
10. Graham, D. B., Allinson, N. M.: Characterizing Virtual Eigensignatures for General Purpose Face Recognition, (in) Face Recognition: From Theory to Applications, NATO ASI Series F, Computer and Systems Sciences, vol. 163. H. Wechsler, P. J. Phillips, V. Bruce, F. Fogelman-Soulie and T. S. Huang (eds), (1998), 446-456.
11. Xiong, H., Swamy, M. N. S., Ahmad, M. O.: Two-dimensional FLD for face recognition, Pattern Recognition, vol. 38, (2005), 1121-1124.

An Improved Representation of Junctions Through Asymmetric Tensor Diffusion

Shawn Arseneau and Jeremy R. Cooperstock

Centre for Intelligent Machines
McGill University, Montreal, Canada
{arseneau, jer}@cim.mcgill.ca

Abstract. Junctions form critical features in motion segmentation, image enhancement, and object classification to name but a few application domains. Traditional approaches to identifying junctions include convolutional methods, which involve considerable tuning to handle non-trivial inputs and diffusion techniques that address only symmetric structure. A new approach is proposed that requires minimal tuning and can distinguish between the basic, but critically different, 'X' and 'T' junctions. This involves a multi-directional representation of gradient structure and employs asymmetric tensor diffusion to emphasize such junctions. The approach combines the desirable properties of asymmetry from convolutional methods with the robustness of local support from diffusion.

1 Introduction

Extracting high-level structure from image gradients is central to many computer vision applications such as data interpolation, 3D scene reconstruction, image enhancement, motion segmentation, and biometrics. Each requires a local description of structure that includes contours and junctions. For example, processing laser-rangefinder data or a stereo-depth map may involve interpolation of sparse data, minimizing the effects of noise, and segmenting this information into distinct objects [1]. Similar issues are found in the context of image enhancement as contours and junctions denote regions where smoothing should be inhibited [2]. In motion segmentation, identifying junctions in the spatio-temporal domain indicate points of occlusion in a video sequence [3]. Similarly, junctions are used to determine salient keypoints in fingerprints or defects in lumber [4,5].

One approach to highlighting junctions is to apply *diffusion*, where local information is distributed to its neighbors conditioned on specific parameters and replaced with the consensus from that data. For example, isotropic diffusion applies local averaging, weighted by relative proximity, to produce a blurring effect. Most diffusion methods make use of gradient information represented by a structure tensor [6]. Although it has several benefits, the structure tensor is limited in that it may only represent gradient in a *symmetric*, or π-periodic form. This implies that diffusion using such a form also results in symmetric information, thus preventing the distinction between 'X' and 'T' junctions from being made. A method to convert this symmetric information into a richer asymmetric form had yet to be incorporated into the diffusion framework. A novel, two-step solution is presented in Section 3, which first transforms the symmetric gradient information into a directional voting field representation and second, iteratively applies asymmetric tensor diffusion. The results of this approach are evaluated experimentally and contrasted with various competing methods in Section 4. Finally, several applications and future work are discussed in Section 5. Before describing the details of our approach, some terminology is defined and a brief review of previous convolution and diffusion approaches is provided in Section 2.

2 Background

To motivate this work, we begin with a review of convolution-based and diffusion approaches with an emphasis on junction analysis. For consistency of terminology, we refer to *direction* as the angle of a vector with respect to the x-axis, ranging between $[0, 2\pi)$ while *orientation* is π-periodic ranging between $[0, \pi)$. Symmetry in our work refers to the geometric interpretation with respect to gradient-based contours and does not refer to the concept of symmetric matrices.

2.1 Convolution-Based Approaches

Early work in the area of convolution-based, directional distribution functions (DDFs) began with the use of Gabor filters [7]. The DDF is created by convolving rotated versions of the kernel at discretely sampled angles and incrementing their respective, angular bins similar to orientation histograms [8]. (examples of DDFs are shown in figure 3) Although the Gabor uses a quadrature pair to address both even and odd-phased gradients, its form is symmetric thus preventing the distinction between 'T' and 'X' type junctions directly. Asymmetric kernels were proposed to highlight such distinctions. For example, Gaussian derivatives were used to derive logical/linear operators and one-sided filter pairs [9,10], while later work by Simoncelli and Farid improved on the accuracy by designing a set of polar-based Gabor kernels, known as wedge-filters [11].

The DDF maxima for these methods do not necessarily imply gradient structure along the direction of the maxima as they are template-matching approaches at their core: implying that they are best suited to finding matches between patterns and not necessarily designed to identify gradient structure [12]. Although steerability has been explored in the use of such approaches, [10], they also require a large bank of filters to address different spatial frequencies [13]. Improved results were obtained using the rotated averaging wedge method (RWAM), which calculated average pixel values within wedge-shaped regions and generated the DDF as the 1D derivative of these values [14]. More recent work by Michelet et al. used a homogeneity function based on an asymmetric sampling grid to populate the DDF, albeit without the benefit of local support through diffusion and also requiring considerable parameter tuning [15]. Although these approaches perform well on trivial, step-edge images, they are inappropriate for estimating gradient direction on more complex data such as that in Figure 3a.

2.2 Diffusion-Based Approaches

The convolution-based approaches described in section 2.1 apply a kernel to the data at a given scale to create the DDF. An alternative is to propagate gradient information from all pixels to their corresponding neighbors. This process, known as diffusion or regularization, maintains a balance between the original information through the data consistency term, while biasing the local model using the diffusion term [2]. The gradient data is best represented using a structure tensor as it encodes not only the orientation and magnitude but the coherence, as in Equations (2-4):

$$S = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \tag{1}$$

$$\theta = \tan^{-1}\left({}^{\boldsymbol{e}_{1y}}\!/\!_{\boldsymbol{e}_{1x}} \right) \tag{2}$$

$$|S| = (\lambda_1 - \lambda_2) \tag{3}$$

$$\varsigma = \begin{cases} \left(\frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}\right)^2 & if \;\; (\lambda_1 + \lambda_2) > 0 \\ 0 & otherwise \end{cases} \tag{4}$$

where I_x is the partial derivative of image $I(x, y)$ with respect to x, ($\boldsymbol{e}_1$, $\boldsymbol{e}_2$,λ_1, λ_2) represent the eigenvectors and values from the decomposition of structure tensor S respectively and ς is the coherence measure as outlined by Jähne that provides a measure of certainty in gradient direction along $\boldsymbol{e}_1$ with respect to $\boldsymbol{e}_2$ [6].

Diffusion techniques can take many forms. For example, isotropic diffusion is a common approach where data is propagated to its neighbors based solely on relative proximity. This method reduces noise at the expense of maintaining high gradient (edge) information. Anisotropic diffusion, reviewed in detail by Tschumperlé and Deriche [2], preserves edges by restricting smoothing across high gradient regions [16].

Orientation diffusion, enforces the periodic nature of symmetric gradient information through the use of a specialized influence function [17]. Several other works refer to direction-based diffusion in the context of gradient polarity direction to bolster pixel-based feature points or in the framework of color enhancement [18,19]. However, the focus of our work is on a phase-*independent* description of the gradient structure.

To account for more complex interaction between gradient structures, tensor voting was introduced [1,20]. It not only diffuses based on proximity, but also on relative curvature as well as allowing for slowly-varying, orientational patterns with a biasing parameter to favor linear rather than curved contours. This approach is adept at handling both sparse and noisy data and requires minimal memory requirements by using a single tensor representation at each node. Relaxation labeling, which permits multiple representations [21], adds support to those pixel locations or *nodes* that have compatible structures based on such criteria as co-circularity, co-heliocity or the normal and tangential curvature components to the tensor fields [22,21,23].

3 Asymmetric Tensor Diffusion (ATD)

The previously described diffusion-based approaches directly apply symmetric information derived from the structure tensor, implying that the resulting DDFs are also symmetric. This section proposes a technique to transform the orientation of the structure tensor data into a directional-based voting field to allow for an asymmetric form at each node. Specifically, a voter distributes ballots to all of its neighbors (receivers). The ballots are collected into their respective DDFs, which are then used to seed a secondary, diffusion stage.

3.1 Stage One: Directional Voting Field

Early work in the conversion of orientation data into a meaningful, directional voting field was proposed in [24]. The present work serves not only to clarify and extend this concept but also to provide a means by which to properly diffuse such information. We first determine the orientation, magnitude and coherence of the structure tensor as per Equations (2-4). Next, an *inwardly* facing *directional bin field*, which represents the initial ballots and the spatial locations of their corresponding receivers, is constructed. Each ballot points towards the axis perpendicular to the orientation of the structure tensor, as per Equations 5-6.

$$B_{ij}\left(\theta_i,\varepsilon\right) = \begin{cases} \theta_i + \varepsilon\pi & \left|\theta_i - \varphi_{ij}\right| > \frac{\pi}{2} \\ \theta_i + \left(\varepsilon + 1\right)\pi & otherwise \end{cases} \tag{5}$$

$$\Psi_{ij}\left(\theta_i,\varepsilon\right) = \begin{cases} 1 & -\frac{1}{2}\tau \le x_{ij} \le \frac{1}{2}\tau \\ 1-\varepsilon & otherwise \end{cases} \tag{6}$$

where $\varepsilon=\{0,1\}$, B_{ij} and Ψ_{ij} denote the ballot direction and magnitude [1] respectively, θ_i is the orientation of the structure tensor at i, τ is the minimum distance between nodes, and φ_{ij} denotes the angle from voter i to receiver j with respect to the x-axis. A directional bin field created from a horizontally oriented input is illustrated in Figure 1b, where the ballots point inward towards the vertical axis. To account for the ambiguity in the original orientation, two opposing ballots are placed along the vertical axis, centered on the original data. The ballots are aligned *parallel* to the original orientation, rather than being steered toward the center point to prevent biasing at this early stage of processing.

The strength of each ballot sent from voter i to receiver j is then weighted by an anisotropic map, known as the region-of-influence (ROI) function, Λ_{ij}, aligned with the orientation of the original data:

$$\Lambda_{ij}\left(\theta_i,\varepsilon\right) = G\left(0,\sigma_x\right)\cdot\Psi_{ij}\left(\theta_i,\varepsilon\right)\cdot R\left(\theta_i\right) \tag{7}$$

where G is a 2D Gaussian with zero mean and $\sigma_y = q\sigma_x$ (Figure 1c) where q is the sigma ratio, and $R(\theta_i)$ is the rotation matrix.

[1] In essence, Ψ distinguishes between single- and double-ballot locations where the latter is assigned to points of orientational ambiguity.

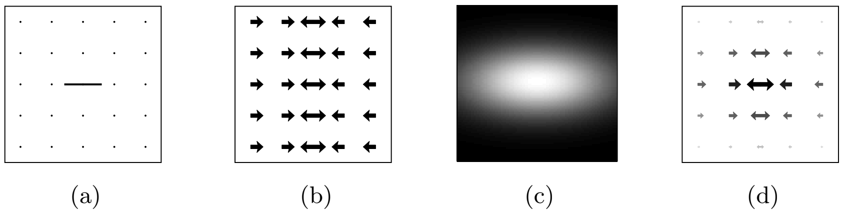


Fig. 1. Stage one: structure tensor with horizontal orientation and coherence=1 (a), B_{ij} (b), Λ_{ij} (c), and the directional voting field where both the vector magnitude and grayscale denote their relative influence (d)

Rather than summing the collected ballots at each receiver into a single value, a histogram of N directional bins is used to collect the ballots, as per Equation 8. While this requires greater memory than that of anisotropic diffusion, tensor voting, and relaxation labeling, it allows for the all-important representation of asymmetric structures.

$$DDF_j\left(\vartheta\right) = \sum_{\varepsilon=0}^{1} \sum_{i=1}^{\Omega} \tilde{S}_i \cdot \Lambda_{ij}\left(\theta_i, \varepsilon\right) \cdot m\left(\vartheta, B_{ij}(\theta_i, \varepsilon)\right) \tag{8}$$

$$m\left(p, q\right) = \begin{cases} 1 & p = q \\ 0 & otherwise \end{cases} \tag{9}$$

where ϑ denotes the directional bin, Ω the local neighborhood around j and $\tilde{S}_i$ the normalized version of the original structure tensor S_i. In brief, voter i sends a ballot $\tilde{S}_i$, weighted by Λ_{ij}, to bin $\vartheta = B_{ij}$ of receiver j.

3.2 Stage Two: Iterative Diffusion of DDF

From stage one, the DDF of each receiver is represented by a single structure tensor per directional bin. This is transformed into a 1D-DDF prior to the diffusion process using a summation of 2π-periodic, normalized Gaussians, $G_{2\pi}(\mu, \sigma, x)$.

$$DDF\left(\vartheta\right) = \sum_{\beta=1}^{N} \left|S_\beta\right| \cdot G_{2\pi}\left(\theta_\beta, \sigma_{\varsigma_\beta}, \vartheta\right) \tag{10}$$

$$\sigma_\varsigma = \left(1 - \varsigma\right)\left(\sigma_{\max} - \sigma_{\min}\right) + \sigma_{\min} \tag{11}$$

Gaussians, normalized to unity area under the curve, are amplified by the magnitudes, $|S_\beta|$, where their means are centered at θ_β and variances are a function of coherence. The values of $(\sigma_{min}, \sigma_{max})$ were assigned empirically as $(0.25, 2)$ to vary between *certain* and *uncertain* estimates, where coherence is bounded between $[0, 1]$. An example of this transformation with three populated bins of successively decreasing coherence, is illustrated in Figure 2. The more elongated ellipses correspond to greater coherence values, which are reflected by Gaussians

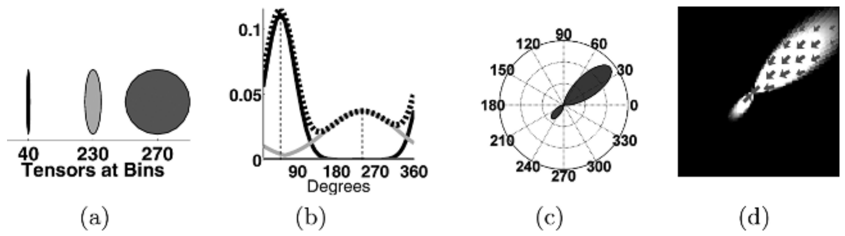


Fig. 2. Stage Two: Three tensors at directional bins 40^o (black), 230^o (light-gray) and 270^o (dark-gray) (a) and their corresponding Gaussians and their 1D-DDF (dashed-line) (b), polar-equivalent (c) and associated weighted, voter-facing, ballot field (d)

of lower variance. These are summed to form the 1D-DDF, which is then converted into a 2D weighting map with radial-based decay as per Equation 12 and illustrated in Figure 2d.

$$\hat{\Lambda}_{ij} = \begin{cases} 1 & \rho_{ij} < DDF\left(\varphi_{ij}\right) \\ \cos\left[\frac{\pi}{2}\left(\frac{\rho_{ij}-DDF(\varphi_{ij})}{\rho_{\max}-DDF(\varphi_{ij})}\right)\right] & DDF\left(\varphi_{ij}\right) \leq \rho_{ij} \leq DDF\left(\varphi_{ij}\right) + \rho_{\max} \\ 0 & otherwise \end{cases} \tag{12}$$

where the hat notation of $\hat{\Lambda}$ reflects the second stage, ρ_{ij} is the Euclidean distance between nodes i and j, and $\rho_{\max}$ denotes the degree of radial decay. Here, each receiver j, obtains a single ballot from voter i, corresponding to the DDF value at i pointing towards j. This ballot increments a directional bin at j that points towards i. We refer to this arrangement as the *voter-facing ballot field.* The DDF is then updated as:

$$DDF_j^t\left(\vartheta\right) = \alpha \cdot DDF_j^{t-1}\left(\vartheta\right) + (1-\alpha) \cdot \sum_{i=1}^{\Omega} \hat{\Lambda}_{ij} \cdot DDF_i^{t-1}\left(\varphi_{ij}\right) \cdot m\left(\vartheta, \left(\varphi_{ij}+\pi\right)\right) \tag{13}$$

where t refers to the iteration step and α denotes the diffusion coefficient. An example of the voter-facing ballot field is illustrated in Figure 2d.

Our approach offers many advantages. First, the only free parameters to tune are the sigma ratio q for G (empirically set to 1/2) and the scale that dictates the range over Ω. Second, by diffusing non π-periodic DDF information, local support is enforced at the pixel level. Finally, this method can also represent endpoints as well as curved structures using the DDF form.

4 Experimental Evaluation

The ATD method is first compared to the convolution approaches against a T-junction image. Next, it is contrasted against the diffusion methods for two tensor field layouts. Finally, ATD is applied to two real-world applications. For all trials and algorithms, N=36.

4.1 Results Against Convolution Approaches

A test-image having asymmetric gradient information derived from several spatial frequencies was used, as shown in Figure 3a. All methods used a scale to match the image of 11x11 pixels. Parameters were tuned for best results against a step-type corner swatch: exemplifying the need for a bank of filters for convolution approaches. The Gabor identified the horizontal orientation; however, was incapable of distinguishing the lack of a downward gradient direction as the filters themselves are symmetric. The one-sided, wedge filter and RWAM fall victim to the presence of several pixel-values along orientations that do not radiate from the center of the image while the ATD properly depicts the three gradient directions.

4.2 Results Against Diffusion Approaches

The test sets used to compare with the previous diffusion methods depict an 'X' and 'T'-shaped structure tensor layout (Figures 4(a,g) respectively). Note that the DDFs shown are from the annotated points 'P' and 'Q'. For the isotropic, anisotropic and tensor voting methods, the DDF is represented by a single structure tensor and visualized as an ellipse oriented along $\boldsymbol{e_1}$ with major and minor radii corresponding to (λ_1, λ_2). From their results, it is not obvious whether it resulted from two, perpendicular tensors, or a single tensor with less certainty without further processing. While relaxation labeling was able to distinguish the two orientations, only the ATD could disambiguate between the two cases.

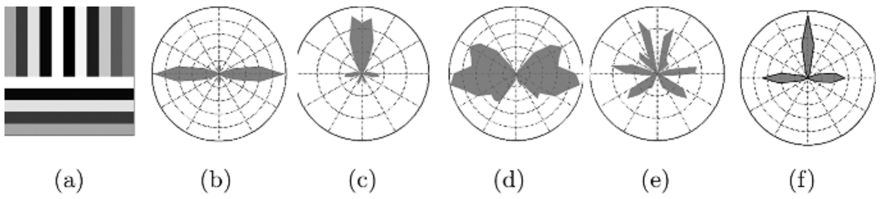


Fig. 3. (a) Input image and resulting DDF's for (b) Gabor, (c) one-sided [10], (d) wedge filters [11], (e) RWAM [14] and (f) ATD

4.3 Results with Real-World Data

Junction structures are key to occlusion detection in video sequences as occlusion is denoted as splitting or merging of contours [3]. Using the *flower garden* sequence, a spatio-temporal slice was extracted and ATD applied. Close-ups of the annotated locations of Figure (5d) correspond to occlusion, disocclusion and no occlusion respectively. The scale used was 9x9 and DDFs reflect three iterations for this experiment.

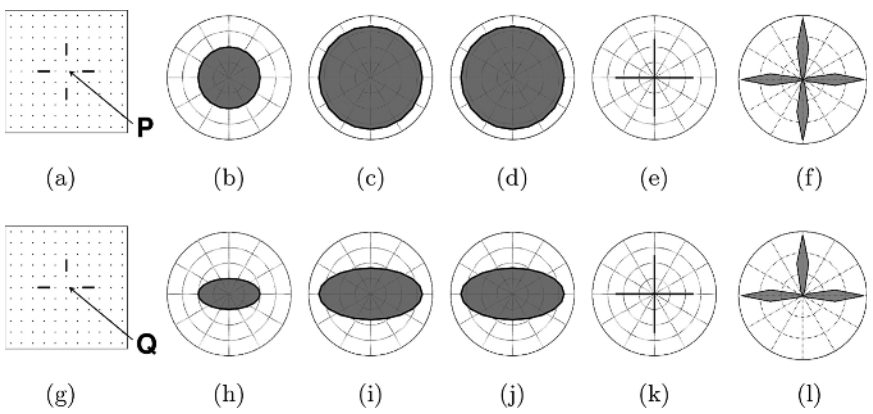


Fig. 4. Top row: X-junction test case and bottom row: T-junction test case. 1st column: initial test layouts, 2nd column: isotropic, 3rd column: anisotropic, 4th column: Tensor Voting [1], 5th column: Relaxation Labeling [22] and 6th column: ATD.

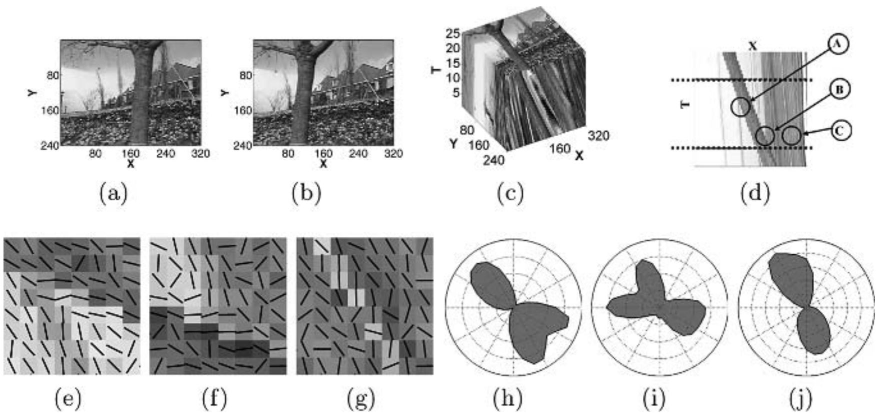


Fig. 5. Two images from the *flower garden* sequence (a,b) with the corresponding spatio-temporal volume (c), where a 2D slice is taken from y=80 (d), indicating three sample locations:'A','B','C' denoting occlusion, disocclusion and no occlusion respectively. The initial gradient orientations (e,f,g) and resulting DDFs from proposed approach (h,i,j) taken from the center of the sample regions.

Junction are also important in fingerprint analysis. A trivial junction model was implemented as a proof-of-concept. For a given DDF, lobe-based features were depicted at DDF maxima with an associated saliency equal to the area of said lobe (integral between the maxima's left and right-wise local minima). Lobes having a saliency of less than 10% of the maximum saliency per DDF were trimmed. Figures (6(b-e)) depict the identification of nodes with a 1,2,3 and 4 lobes respectively.

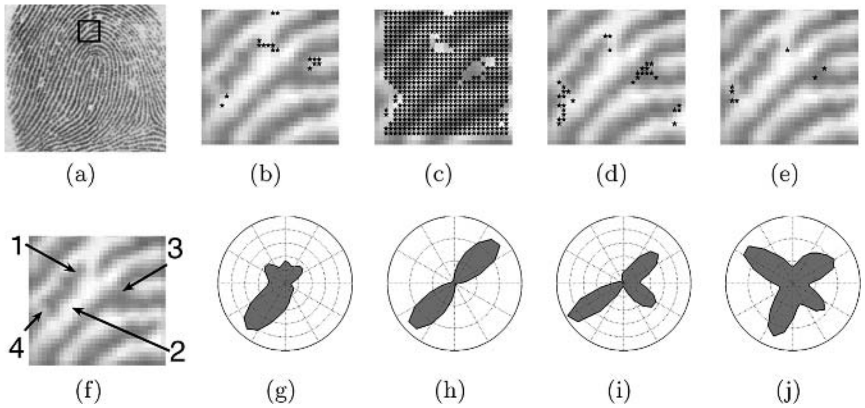


Fig. 6. Fingerprint (a) and its sub-image (f), junction classification for 1,2,3 & 4 lobes (b-e) respectively with DDF close-ups corresponding to labels (1-4)from (f) in (g-j)

5 Conclusion

A novel approach of transforming symmetric gradient information into asymmetric DDFs is proposed, along with a method by which to diffuse them. The accuracy is shown to be an improvement over current convolution-based approaches and the asymmetric representation allows for the distinction between non-π periodic structures. Future work will investigate a multi-scale as well as a 3D implementation.

References

1. Nicolescu, M., Medioni, G.: Motion segmentation with accurate boundaries - a tensor voting approach. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition. Volume 1. (2003) 382–389
2. Tschumperlé, D., Deriche, R.: Diffusion pde's on vector-valued images. IEEE Signal Processing Magazine (2002) 16–25
3. Bolles, R., Baker, H., Marimont, D.: Epipolar-plane image analysis: An approach to determining structure from motion. International Journal of Computer Vision **1** (1987) 7–55
4. Jiang, X.: On orientation and anisotropy estimation for online fingerprint authentication. IEEE Trans. On Signal Processing **53** (2005) 4038–4049
5. Rao, A., Jain, R.: Computerized flow field analysis: Oriented texture fields. IEEE Trans. On Pattern Analysis and Machine Intelligence **14** (1992) 693–709
6. Jähne, B.: Spatio-Temporal Image Processing: Theory and Scientific Applications. Volume 751. Springer-Verlag, Berlin (1993)
7. Adelson, E., Bergen, J.: Spatiotemporal energy models for the perception of motion. Journal of the Optical Society of America **2** (1985) 284–299
8. Freeman, W., Tanaka, K., Ohta, J., Kyuma, K.: Computer vision for computer games. In: 2nd International Conference on Automatic Face and Gesture Recognition. (1996) 100–105

9. Iverson, L., Zucker, S.: Logical/linear operators for image curves. IEEE Trans. on Pattern Analysis and Machine Intelligence **17** (1995) 982–996
10. Perona, P.: Steerable-scalable kernels for edge detection and junction analysis. In: Proc. European Conference on Computer Vision. (1992) 3–18
11. Simoncelli, E., Farid, H.: Steerable wedge filters for local orientation analysis. IEEE Trans. On Image Processing **5** (1996) 1377–1382
12. Malik, J., Belongie, S., Shi, J., Leung, T.: Textons, contours and regions: Cue integration in image segmentation. In: Proc. IEEE International Conference on Computer Vision. Volume 2. (1999) 918–925
13. Heeger, D.: Optical flow using spatiotemporal filters. International Journal of Computer Vision **1** (1988) 279–302
14. Yu, W., Daniilidis, K., Sommer, G.: Rotated wedge averaging method for junction characterization. In: Proc. IEEE Computer Vision and Pattern Recognition. (1998) 390–395
15. Michelet, F., Germain, C., Baylou, P., Costa, J.D.: Local multiple orientation estimation: Isotropic and recursive oriented network. In: Proc. IEEE International Conference on Pattern Recognition. Volume 1. (2004) 712–715
16. Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. IEEE Trans. Pattern Anal. Machine Intell. **33** (1990) 629–639
17. Perona, P.: Orientation diffusions. IEEE Trans. On Image Processing **7** (1998) 457–467
18. Tang, B., Sapiro, G., Caselles, V.: Direction diffusion. In: International Conference on Computer Vision. (1999) 1245–1252
19. Lowe, D.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60** (2004) 91–110
20. Tong, W., Tang, C., Mordohai, P., Medioni, G.: First order augmentation to tensor voting for boundary inference and multiscale analysis in 3d. IEEE Trans. On Pattern Analysis and Machine Intelligence **26** (2004) 594–611
21. Parent, P., Zucker, S.: Trace inference, curvature consistency, and curve detection. IEEE Trans. On Pattern Analysis and Machine Intelligence **11** (1989) 823–839
22. Ben-Shahar, O., Zucker, S.: Hue fields and color curvatures: A perceptual organization approach to color image denoising. In: IEEE Computer Vision and Pattern Recognition. Volume 2. (2003) 713–720
23. Savadjiev, P., Campbell, J., Pike, G., Siddiqi, K.: 3d curve inference for diffusion mri regularization. In: International Conference on Medical Image Computing and Computer Assisted Intervention. (2005) 123–130
24. anonymous for peer review: An asymmetrical diffusion framework for junction analysis. In: British Machine Vision Conference (to appear). (2006)

Accurate Extraction of Reciprocal Space Information from Transmission Electron Microscopy Images

Edward Rosten and Susan Cox

Los Alamos National Laboratory, Los Alamos, NM, USA
{edrosten, scox}@lanl.gov

Abstract. As the study of complex systems has become dominant in physics the link between computational and physical science has become ever more important. In particular, with the rising popularity of imaging techniques in physis, the development and application of cutting edge computer vision techniques has become vital. Here we present novel image analysis methods which can be used to extract the position of features in diffraction patterns (reciprocal space) with unprecedented accuracy.

The first contribution we have developed is a method for calculating the nonlinear response of photographic film by using the noise in the image enabling the extraction of accurate intensity information. This allows high-resolution (but non-linear) film to be used in place of low-resolution (but linear) CCD cameras. The second contribution is a method for accurately localising very faint features in diffraction patterns by modelling the features and using the expectation maximization algorithm directly on the image to fit them. The accuracy of this technique has been verified by testing it on synthetic data.

These methods have been applied to transmission electron microscopy data, and have already enabled discoveries which would have been impossible using previously available techniques.

1 Introduction: TEM and the Importance of Image Analysis

Over the last twenty years the ability to image materials with electrons, which have a wavelength considerably smaller than light, has revolutionised the physical sciences. However, generating meaningful data from the images obtained often requires automated image analysis. A number of programs have been created which perform basic operations well, but when dealing with diffraction patterns (reciprocal space) and with situations where accuracy and statistical significance are critical these are not adequate. Here we present methods for extracting extremely accurate and statistically significant data from diffraction patterns.

The diffraction patterns shown here were obtained using transmission electron microscopy (TEM), an important technique in both academia and industry. In a transmission electron microscope [1] a beam of electrons accelerated through

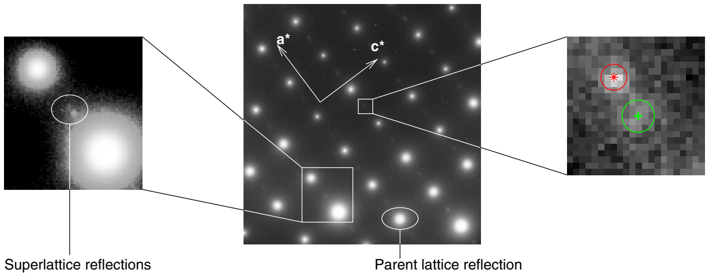


Fig. 1. A diffraction pattern taken from $La_{0.5}Ca_{0.5}MnO_3$. The parent lattice reflections are caused by the cubic parent lattice. The superstructure reflections are caused by the additional ordering which occurs at $\sim$ 220K [2], and occur along the primary axis (labelled a^*). The cutout has been enhanced to make the weak reflections more visible, since they are present as very faint spots. The superlattice reflections centred at $*$ and $+$ are shown in the other highlighted area of the diffraction pattern. The positions are found using the method described in Section 4.

a high voltage passes through a thin ($<$ 200 nm) area of a sample. The beam then passes through a series of magnets which act as lenses. By adjusting the strength of these magnets either an image of the sample (real space information) or a diffraction pattern (reciprocal space information) can be observed, and recorded using photographic film or a CCD (depending on the microscope). Thus both images and diffraction patterns can be obtained from the same area of the sample. The most advanced TEMs can perform imaging almost at the atomic scale.

A diffraction pattern is essentially a power spectral density of an affine projection of the crystal lattice (the array of repeating units, or unit cells, which comprise the crystal), with some additional effects arising from dynamical (inelastic) diffraction, which can be ignored here. The main crystal lattice (the *parent lattice*) gives rise to a regular grid of spots termed *parent lattice reflections*. In the case of a typical pseudo-cubic parent lattice (see Figure 1), the grid is described by the two wavevectors a^* and c^*. A wavevector is simply a vector defined in reciprocal space. Some materials have a repeating superstructure with a period greater than the unit cell size superimposed on the parent lattice. In reciprocal space this gives rise to reflections at wavevectors smaller than the unit cell (superlattice reflections). In many strongly correlated systems a one-dimensional electronic superstructure forms at low temperatures. Pairs of superlattice reflections appear along one axis in the diffraction pattern. The position of the superlattice reflections is given by a wave vector whose magnitude is denoted q (with units m^{-1}), though it is often given in units of a^*. The relative intensity of the superlattice and parent lattice reflections varies widely depending on the type of sample, and in some experiments the superlattice reflections can be extremely faint as demonstrated by the diffraction pattern for an $La_{0.5}Ca_{0.5}MnO_3$ thin film shown in Figure 1.

Although a great deal of effort and expertise has gone into designing packages for the analysis of certain kinds of TEM images (e.g. SEMPER [3], Digital Micrograph), analysis of diffraction patterns has typically been performed by hand, which is time consuming and inaccurate. In Section 2 we present a technique for finding the nonlinear response of photographic film, which allows analysis techniques which require a linear response to be used. In Section 4, we present a system which not only allows individual measurements of superlattice reflections to be made with high, and quantifiable, accuracy, but can measure the position of every reflection in a diffraction pattern. This allows results to be measured to a high level of statistical significance, and makes it feasible to obtain data from patterns which would previously have been abandoned as unusable. These techniques have been applied to previously unsolvable problems and the results are presented in Section 5.

2 Nonlinear Film Response Correction

CCD image sensors have a linear response with electron intensity which allows the intensity of the image to be accurately measured. However, the use of photographic film has two advantages relative to using a CCD: firstly the resolution of scanned film is high (typically $\sim 4000 \times 4000$ pixels, compared to 512×512 for common TEM CCD sensors); secondly, the physical robustness of the film allows much longer exposure times (excessive illumination of a CCD with electrons causes damage). This heavily saturating the brighter reflections, allowing the faint ones to be visible. However, in order to interpret intensity information from photographic film correctly it is necessary to process the image, so that the image intensity is proportional to electron intensity.

Manufacturers provide calibration curves for the film, but they are only accurate if development conditions are identical. This is unlikely to be the case since the strength of the developing solutions changes over time as they are used or replaced. Instead, we present a method where the film response can be deduced from the image noise, individually for each image.

Noise in the electron intensity is approximately constant over the image, and is caused by random scattering of the diffracted electrons (the diffraction patterns are typically not scanned at a resolution where film grains are visible, and the level of shot noise is negligable, since the exposure time is long). Therefore, at a given film intensity, f_i, the amount of image noise, η, is related to the electron intensity e_i:

$$\eta \approx \frac{\mathrm{d}\, f_i}{\mathrm{d}\, e_i}, \tag{1}$$

since $\frac{\mathrm{d}\, f_i}{\mathrm{d}\, e_i}$ is the film sensitivity and the electron intensity can be arbitrarily set so that the noise variance is unity. We use this relationship to find the response of the film. To do this, we first smooth the image, to remove noise and then find the difference between the smoothed image ($\hat{f_i}$) and the unsmoothed image (f_i) at every pixel (the difference being due to noise). The measurements are binned by image intensity, and the amount of noise is measured by taking the standard

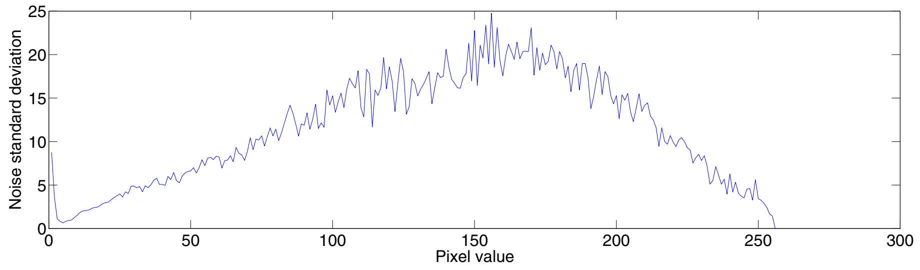


Fig. 2. Plot of noise intensity against image intensity, taken from a typical diffraction pattern

deviation of the measurements in each bin. Linear smoothing techniques such as convolving with a Gaussian kernel are unsuitable because they cause features to spread out. Instead, we fit polynomials of order r to groups of n consecutive pixels along every scanline to smooth the image. The values of n and r depend loosely on the feature size.

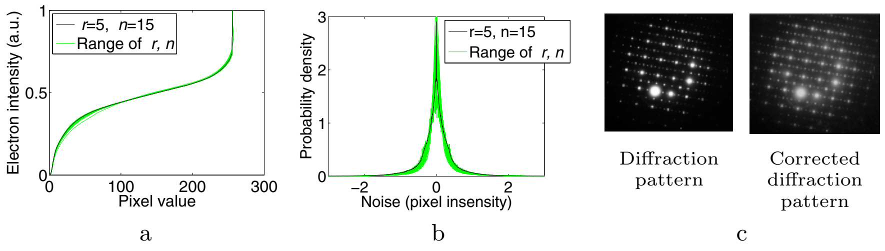


Fig. 3. (a) Film response measured with r in the range $3 \ldots 10$ (in steps of 1) and n in the range $3r \ldots 50$ (in steps of 10). (b): Noise distributions for the film responses. (c) A diffraction pattern before and after correction with the film response curve, using $r = 5$ and $n = 15$.

Having measured η for all image intensities, we now have an approximate measure of the differential of the film response, which is shown in Figure 2. Note however the increase in noise for very small pixel values. This is an artifact caused by the the image being clipped at an intensity of zero, which is not taken in to account by the polynomial fitting (even though after fitting, the polynomial is clipped). The function mapping the film intensity to the electron intensity, $e_i = \rho(f_i)$ is given by:

$$\rho(f_i) = \int_0^{f_i} \frac{1}{\eta(\tau)} \mathrm{d}\,\tau \tag{2}$$

and

$$\rho(0) = 0. \tag{3}$$

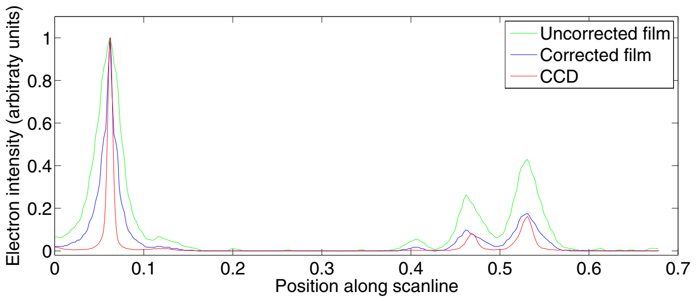


Fig. 4. Plots of scanlines through diffraction patterns of $La_{1-x}Ca_xMnO_3$. The plot shows that the calibrated film response and CCD response are very similar. The differing peak widths are caused by differences in focus.

Using this equation on the data from Figure 2, we obtain the response curve ρ, shown in Figure 3 a. The integral relationship is useful since it results in a curve which is significantly smoother than the measured noise data. Is should be noted that result is not well modelled by a classic gamma correction curve ($\rho(i) = i^\gamma$): the curve for low pixel values is well modelled in this manner, but the response at high pixel values is dominated by the film saturating. The result of applying this to an image is shown in Figure 3 c.

Figure 3 a, b also show the effect of different values of r and n. As can be seen, the computed response is insensitive to these values. Further, it shows that the noise distribution is strongly non-Gaussian. This backs up the calculation showing that the shot noise is small, since at the currents and exposure times used, the distribtuion of shot noise (which follows a Poisson distribtuion) would be approximately Gaussian.

2.1 Evaulation

In order to test the film calibration, one would ideally expose the film to a known pattern (such as a ramp), and record the film response to the known electron intensity. Unfortunately, this is not possible with the equipment involved. If a sample viewed with a CCD is compared to a sample viewed on film, then the relative intensities of the spots should be the same (when the film is corrected for its nonlinear response). This is because a CCD responds linearly to electron intensity.

Figure 4 shows linescans through a diffraction pattern taken on a CCD and film. For comparison, the background level has been removed, and the brightness of the images adjusted so that the main peaks are of the same intensity.

The darker spots are significantly stronger on the film than on the CCD. When the film is corrected, the spots are approximatly the same height. This gives a good indication that the film calibration produces accurate results.

3 Accurate Measurement of Parent Lattice Reflections

In order to measure the superlattice reflections, the parent lattice has to be found first. This is done by manually identifying two adjacent parent lattice reflections. The positions of these are then refined using the mean-shift algorithm [4]. Since the diffraction pattern is regular, two spots are enough to define the entire grid of reflections. In particular the pair of spots are used to define the primary (a^*) axis, which is the direction along which the superlattice ordering occurs. However, using only two spots to define the grid is not sufficiently accurate without further refinement.

The two parent latice positions defines the entire grid of reflections. The distortion is sufficiently low that each point lies close enough to the relavent real parent lattice reflection that mean-shift can successfully refine the position.

There currently exist a variety of techniques for finding the parameters of imaging systems, such as [5], which finds the parameters of pinhole cameras, and [6] which find the parameters of a more sophisticated model, which models nonlinearities with radial distortion. However, not only is the imaging system in a TEM is not well modelled by these, but many of the standard calibration procedures require multiple views of a 3D scene. Instead, we have a set of correspondences between points in the image (the lattice reflections) and a known shape (a 2 dimensional square grid, the scale, position and orientation of which can be chosen arbitrarily), so we use a general purpose distortion model. The model we use to cover all the distortions is akin to a nonlinear (higher order) version of the homography (this is similar to the model presented in [7]):

$$\begin{pmatrix} sx \\ sy \\ s \end{pmatrix} = \begin{pmatrix} h_{1,1} & \dots & \dots & & h_{1,2n+1} \\ h_{2,1} & \dots & \dots & & h_{2,2n+1} \\ 0 \;\dots\; 0 & h_{3,2n-1} & h_{3,2n} & & 1 \end{pmatrix} \begin{pmatrix} X^n \\ Y^n \\ \vdots \\ X \\ Y \\ 1 \end{pmatrix}, \tag{4}$$

where $(x\ y)$ are the grid coordinates and $(X\ Y)$ are the image coordinates, normalised so that the range of X and Y is ± 1. Writing $\mathbf{H} = \left(\begin{smallmatrix} h_{1,1} & \cdots \\ \cdots & 1 \end{smallmatrix}\right)$, we first find the parameters of $\mathbf{H}$ using a linear solution and then refine this using reweighted least-squares to minimize the image-space error. Reweighting is required because despite the distortion model, some errors are not easily modelled. For instance, a relatively large amount of distortion can occur nearby where the film is clamped in the scanner. Apart from the components of $\mathbf{H}$ required to get the image at the correct orientation, position and scale, the components are typically quite small, and make corrections on the order of 2–3 pixels towards the edge of the image.

4 Accurate Localisation of Superlattice Reflections

In order to find the superlattice reflections, several standard feature detectors were tested, Harris [8], a DoG (difference of Gaussians—the detector used for SIFT [9] features) based detector and FAST [10]. However, due to the noisy background and the faint, sometimes overlapping nature of the features, the performance of these detectors was poor on all but the best images. Instead, a model of the image (as opposed to a general purpose detector) is used to achieve much more accurate detection.

A diffraction pattern can be considered to be an unnormalised probability distribution, with each pixel representing the probability that a given diffracted electron will end up there. In addition to linear diffraction, the electrons also undergo several linear and nonlinear effects. The end result is that the superlattice reflections end up approximately Gaussian. There is also spreading from the main central reflection, which results in the space between the spots not being completely black. This spreading typically takes the form of a very shallow gradient away from the central reflection. On the scale of a pair of superlattice reflections, this can considered to be flat. Therefore, we can model the patch of image around a pair of superlattice reflections as a mixture model consisting of two isotropic Gaussians (the superlattice spots) and a constant background level, where the size, position and scale of the Gaussians and the scale of the background are the degrees of freedom. We can then fit this model direcly to the image patch around the superlattice reflections using the Expectation Maximization (EM) algorithm [11,12]. Since a reasonable initialization for EM is available—a^* is known and the wave vector (q/a^*) is always quite close to $0.5a^*$—the resulting algorithm is very robust and is capable of finding the positions of very faint spots, in high noise images. This is illustrated in Figure 1.

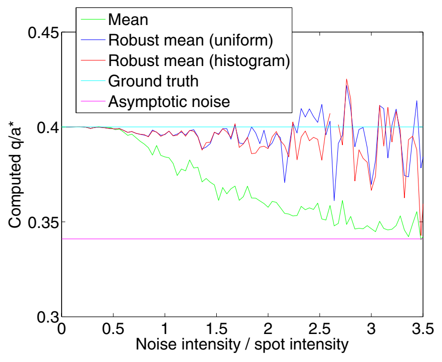
4.1 Evaluation

In order to test the accuracy of the superlattice localisation, the system was run on simulated TEM images so that a ground truth measurement for the spot position was known. The simulations measure accuracy under the following common noise conditions:

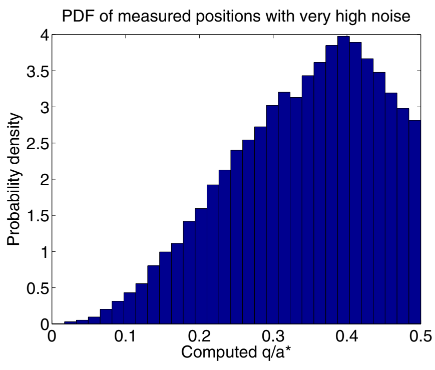
1. Addition of uncorrelated noise to the image.
2. Addition of correlated, unmodelled intensity changes. These occur as a result of spreading of bright parent lattice reflections and are manifested as an intensity ramp aligned with the approximate direction of the superlattice reflections. They are modelled here as a linear ramp, but in practice, the profile is quite variable.

This is tested in the following manner:

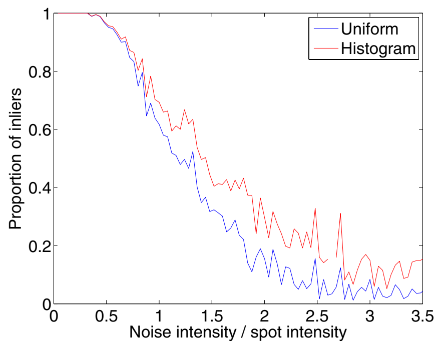
1. Generate ideal image (a constant value with two Gaussian spots at known positions).
2. Add a linear ramp (from $-R$ to R in intensity) aligned with the superlattice reflections.



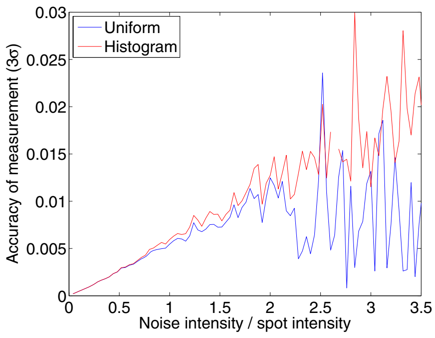
a



b



c



d

Fig. 5. (a) Graph showing how computed values for q/a^* vary with increasing noise. The background intensity is 0.1, the spot intensity is 0.1 (typical values from images) and the noise intensity (standard deviation) varies from 0 to 0.35. The patch size is 31 × 31 pixels. (b) Histogram of computed positions. The mean of this distribution is the asymptotic noise mean in (a). (c) Proportion of points which converge on the inlier (as opposed to background) distribution. (d) Accuracy of spot position, when 300 measurements are taken (a typical number).

3. Add uncorrelated noise, of standard deviation σ to each pixel.
4. Clip image intensities to the range 0–1.
5. Attempt to find the spot positions using EM.

When the mixture model fails to converge to a sensible value (for instance when the size of the spots becomes zero, or the computed position or the spots cannot be correct given that $q < 0.5a^*$) the result for that individual computation is rejected.

The results of the computed mean for additive noise only (i.e. $R = 0$) are shown in Figure 5 a. As can be seen, if the mean value of the computed values is taken, then the mean decreases with increasing additive noise. The reason for this is that EM will either converge on the correct spot position (with some small amount of noise), or some a random position with the 'background' distribution. When the noise gets large, the background distribution will dominate, so the mean computed

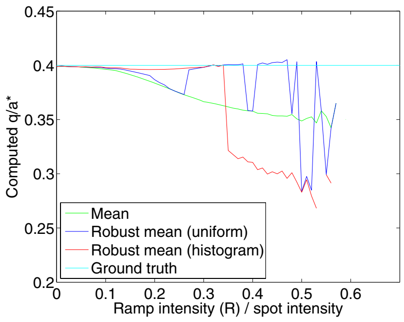


Fig. 6. Plot of the computed spot position against the additive ramp intensity. The parameters are given in Figure 5 and the noise intensity is 0.05. The histogram is taken from the high noise limit with no additive ramp. When the ramp intensity exceeds about 0.6, there are no instances in which the system converges to sensible value.

position will simply be the mean of the background distribution. At intermediate points, the computed mean will be between the two distributions.

The mean can instead be computed robustly by modelling the computed position as a mixture model of a Gaussian (representing correct points) and outliers uniformly distributed between 0 and 0.5. This mixture model can be fitted to the results (using EM), and the mean position of the Gaussian can be taken as the computed spot position. It turns out that the background distribtuion is not in fact uniform as one might expect, but is instead given by the histogram in Figure 5 b. The reason for this is because there is a background constant component in the image which will slightly bias the mixture model to converge towards the centre. However, as can be seein in Figure 5 a, using this distribution, instead of the uniform distribution in the computation of the mean does not produce significantly better results. However, both robust techniques produce significantly better results than the simple mean computation.

The robust computation of the mean also gives the probability that the measurement is drawn from the Gaussian (foreground) or background distribution. This can be used to compute the proportion of measurements which converge to the correct place. This is shown in Figure 5 c. This can also be used to estimate the accuracy (standard deviation) of the computed mean spot position as shown in Figure 5 d. Note that when the noise gets large, the robust mean using the histogram as the background distribution produces considerably more accurate estimates of the accuracy.

For the correlated noise, the results of the computed mean varying with R are shown in Figure 6. This shows that the system is very sensitive to unmodelled correlated noise. This justifies the decision to treat each pair of supperlattice reflections seperately, as opposed to computing the average pixel values over all pairs, since a few patches with strongly correlated noise could easily prevent the system from finding the correct spot position.

5 Conclusions

This paper has demonstrated that it is possible to correct the nonlinearity of photographic film, and to extract the positions of superlattice reflections to a very high degree of accuracy, even when the signal to noise ratio is low. These techniques have allows us to extract information from very faint superlattice reflections. By analysing subtle spatial variations of the superstructure of $La_{0.5}Ca_{0.5}MnO_3$ using the methods presented here, it was possible to show that the periodicity of the superlattice can be altered by altering the strain present in the thin film [13].

References

1. D.B. Williams, B.C.: Introduction to Transmission Electron Microscopy. Plenum Publishing, New York (1996)
2. C.H. Chen, S.-W. Cheong: Commensurate to incommensurate charge ordering and its real-space images in $La_{0.5}Ca_{0.5}MnO_3$. Phys. Rev. Lett. **76** (1996) 4042–4045
3. W.O. Saxton: Semper: Distortion compensation, selective averaging, 3-d reconstruction, and transfer function correction in a highly programmable system. J. Struct. Biol. **116** (1996) 230–236
4. Comaniciu, D., Meer, P.: Mean shift analysis and applications. In: 7th IEEE International Conference on Computer Vision, Kerkyra, Corfu, Greece, Springer (1999) 1197
5. Faugeras, O.D., Luong, Q.T., Maybank, S.J.: Camera self-calibration: Theory and experiments. In: 2nd Euproean Conference on Computer Vision, Springer (1992) 321–334
6. Stein, G.: Lens distortion calibration using point correspondences. In: 11th IEEE Conference on Computer Vision and Pattern Recognition, Springer (1997) 602–608
7. Claus, D., Fitzgibbon, A.W.: A rational function lens distortion model for general cameras. In: 18th IEEE Conference on Computer Vision and Pattern Recognition, Springer (2005) 213–219
8. Harris, C., Stephens, M.: A combined corner and edge detector. In: Alvey Vision Conference. (1988) 147–151
9. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60** (2004) 91–110
10. Rosten, E., Drummond, T.: Machine learning for high speed corner detection. In: 9th Euproean Conference on Computer Vision, Springer (2006)
11. Dempster, A., Laird, N., D.B., R.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society **B 39** (1977) 1–38
12. Redner, R.A., Walker, H.F.: Mixture densities, maximum likelihood, and the EM algorithm. SIAM Review **26** (1984) 195
13. Cox, S., Rosten, E., Loudon, J.C., Chapman, J.C., Kos, S., Calderon, M.J., Kang, D.J., Littlewood, P.B., Midgley, P.A., Mathur, N.D.: Control of $La_{0.5}Ca_{0.5}MnO_3$ superstructure through epitaxial strain release. Physics Review B. (2006)

GPU Accelerated Isosurface Extraction on Tetrahedral Grids

Luc Buatois[1,2], Guillaume Caumon[1,3], and Bruno Lévy[2]

[1] Nancy Université, Gocad Research Group, Nancy, France
Luc.Buatois@gocad.org
[2] ALICE, Inria Lorraine, Vandoeuvre, France
[3] CRPG-CNRS, Vandoeuvre, France

Abstract. Visualizing large unstructured grids is extremely useful to understand natural and simulated phenomena. However, informative volume visualization is difficult to achieve efficiently due to the huge amount of information to process. In this paper, we present a method to efficiently tessellate on a GPU large unstructured tetrahedral grids made of millions of cells. This method avoids data redundancy by using textures for storing most of the needed data; textures are accessed through vertex texture lookup in the vertex shading unit of modern graphics cards. Results show that our method is about 2 times faster than the same CPU-based extraction, and complementary with previous approaches based on GPU registers: it is less efficient for small grids, but handles millions-tetrahedra grids in graphics memory, which was impossible with previous works. Future hardware evolutions are expected to make our approach much more efficient.

1 Introduction

1.1 Motivations

Visualizing isosurfaces is essential in many different fields of scientific research like Computational Fluid Dynamics (CFD), finite element modeling and medical and seismic tomography. These applications often use large unstructured grids made of millions of tetrahedra. Handling these kind of very large grids without out-of-core or parallel algorithms using a simple pc leads our approach.

Each cell, in an unstructured tetrahedral mesh, has a constant topology. Hence, very specialized algorithms for this type of cells have been carried out using hardware acceleration techniques [1,2,3,4]. Unfortunately, these methods introduce sometimes redundancy in the storage method, strongly limiting the size of the grid or, sometimes, use special non-standard functionalities implemented on few graphics cards.

We propose a way to efficiently extract an isosurface from a scalar field by means of modern GPUs for unstructured tetrahedral meshes. Our method handles very large grids made of millions of tetrahedra by means of common GPUs.

1.2 Previous Work

Isosurface extraction can be optimized by going only through the intersected cells in the grid, and by accelerating the extraction of the isovalue polygon in each intersected cell.

When marching through all cells to extract an isosurface, most of the time is spent checking for non-intersected cells. Therefore, algorithmic acceleration techniques have been investigated to discard non-intersected cells before rendering them. Contour seeds [5,6,7] start from seed cells to propagate over the grid to build the requested isosurface. Interval Trees [8] work on value space to classify cells within intervals of values. Octrees [9] recursively subdivide space remembering at each stage the interval of values contained in each subdivision. Others important algorithmic acceleration techniques exist [10,11,12] and are very efficient.

This article is mainly focused on cell tessellation of large grids. Cell tessellation has been extensively studied in the literature for the past 20 years. The Marching Cubes [13] is one of the first efficient techniques to directly tessellate a hexahedron. Cell projection [14] doing indirect extraction or a technique that turns around faces [15,16,17] using topological links between the vertices/faces/edges of each cell, are others basis of art. Our method is inspired by the Marching Cubes [13], so we now review the main steps of this algorithm.

The Marching Cubes algorithm efficiently tessellates a hexahedron, using two precomputed lookup tables: the table of *edges* contains the numbers of the beginning and ending vertices for each cell edge, and the table of *cases* contains all possible configurations of the isosurface to be extracted. A *plus* (resp. *minus*) is attached to each vertex if the value at this vertex is higher (resp. lower) than the isovalue. Out of the $2^8 = 256$ possible configurations for a hexahedron, the Marching Cubes method takes into account symmetries to reduce the size of the table of *cases* to only 15 entries. Knowing both description tables, extracting an isosurface from a hexahedron is easy:

- From the current hexahedron configuration, compute the index in the table of *cases* as: $index = \sum_{i=0}^{7} (F(x_i) >= w) * (i+1)^2$, where $F(x_i)$ is the value attached to the vertex x_i and w the isovalue.
- Read the table of *cases* at this index to retrieve the list of intersected edges.
- Retrieve the end vertices of each intersected edge using the table of *edges*, and compute the intersection with the isosurface by linear interpolation.

The Marching Cubes can be adapted to tetrahedral grids (Marching Tetrahedra), using specific tables of *edges* and *cases* for a tetrahedron (Fig1).

Hardware Acceleration Techniques using GPUs. Programmable graphics hardware have opened new perspectives for isosurface extraction. Currently, due to hardware limitations, the only types of grids that have been hardware-accelerated for direct isosurface extraction are unstructured tetrahedral meshes and regular hexahedral meshes. For regular hexahedral grids, isosurface extraction can be achieved using pre-integrated volume rendering [18,19]. These methods volume-render the whole grid using a Dirac opacity transfer function so that only one isovalue is rendered. They have a high computational complexity as compared to polygonal isosurface extraction.

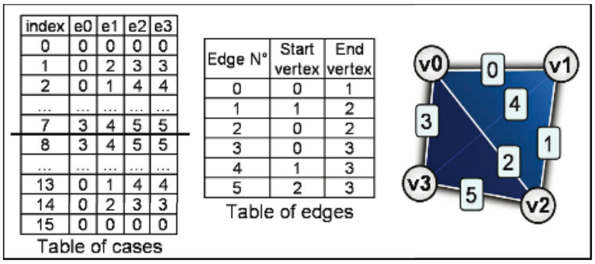
For unstructured tetrahedral meshes, the fastest known GPU hardware accelerated technique was recently introduced by Kipfer et al. [1]. This method is fast, limits data storage redundancy and processing redundancy. However, this method uses the SuperBuffers extension only available on some ATI graphics cards, which are not part of any revision of the Shader Model standard and, therefore, may disappear from the list of supported features on ATI graphics cards. Consequently, we decided to base our work on another more general fast technique shown by Pascucci [2] (also present in other papers [4]). Pascucci proposes a hardware implementation of the Marching Tetrahedra, based on a table of *edges* and a table of *cases*. Its method uses standard features but is only applicable to small and medium-sized grids (less than one million of tetrahedra with 256MB of graphics memory; see Figure 4 for details). Our technique is similar, with the noticeable difference that we completely avoid data redundancy by using indirect indexing.

On a GPU, textures are only reachable on the pixel shading unit for graphics cards supporting Shader Model 2.0, and are reachable in both vertex and pixel shading units for cards fully supporting Shader Model 3.0. Shader Model 3.0 is supported by all Nvidia graphics cards of 6^{th} generation and above; ATI does not currently support this functionality, but claims that its next generation of products will. Previous approaches [2,4] use the vertex shading unit to implicitly extract the isosurface. They send to the GPU four numbered vertices for each tetrahedron, since an iso-polygon in a tetrahedron contains at most four vertices. The number of the iso-polygon vertex, the geometry of the tetrahedron vertices and the corresponding scalar are also streamed to the GPU through variable registers. Constant GPU registers are used to efficiently store the tables of *edges* and *cases*. For each vertex sent to the vertex shader unit, the Marching Tetrahedra algorithm is applied: compute the entry in the table of *cases*, find the edge corresponding to the vertex number (from 0 to 3), seek the extremities of this edge in the table of *edges*, compute the intersection with the isosurface, then move the current vertex to this position. When the iso-polygon is a triangle, the extra vertex sent is stacked at the same place than the last found intersection, and hence is ignored by the graphics card.

It is possible to combine this hardware accelerated isosurface extraction with algorithmic optimization by sending to the GPU only the intersected cells by using, e.g., an Interval Tree [8], an Octree [9] or a Seed Set [5].

1.3 Contributions

Pascucci's method is similar to ours, but one of the strongest bottlenecks of the Pascucci implementation is that sending all required data to the GPU for each tetrahedron introduces a strong redundancy, since most of the vertices are shared between several tetrahedra. The GPU-accelerated method presented in this paper overcomes these limitations by:



index	e0	e1	e2	e3
0	0	0	0	0
1	0	2	3	3
2	0	1	4	4
...	...	...	...	...
7	3	4	5	5
8	3	4	5	5
...	...	...	...	...
13	0	1	4	4
14	0	2	3	3
15	0	0	0	0

Table of cases

Edge N°	Start vertex	End vertex
0	0	1
1	1	2
2	0	2
3	0	3
4	1	3
5	2	3

Table of edges

Fig. 1. Tables of *cases* and *edges*

- avoiding data redundancy through shared vertices
- limiting AGP/PCI-Express bus transfers to a minimum
- efficiently storing the whole data inside a texture to improve isosurface extraction performance for large grids
- handling grids made of millions of cells

Moreover, our method also supports both brute force extraction and combination between GPU tessellation and CPU algorithmic filtering of non-intersected cells like an Interval Tree, an Octree or a Seed Set. In the next section, we describe our method to tessellate tetrahedra in very large grids. For this, we use vertex texture lookup, introduced in the Shader Model 3.0 standard, corresponding to the Nvidia 6^{th} generation (and above) graphics cards.

2 Our Approach

2.1 Preprocessing

Adapted Data Storage for a GPU. The tables of *cases* and *edges* (Figure 1) are defined during the preprocessing stage. The table of *cases* is symmetric, so only the first half of the table must be stored. To handle every possible configuration of the isosurface within a tetrahedron, some edge indexes are duplicated in the table of *cases*. This means that the corresponding vertices will be stacked at the same location and discarded during the real-time rendering stage.

The central question is how to store these two description tables and all required data to efficiently feed a GPU. Previous approaches [2,4] send geometry, values and both tables of *edges* and *cases* through GPU registers. These approaches send for each cell every needed data, and, since most of vertices are shared between several cells, there is a strong redundancy which consumes a lot of graphics memory and calls for more RAM/Graphics-RAM transfers than necessary. Moreover, the whole data is packed in OpenGL vertex arrays or display lists for efficiency, which are also memory hungry. For a tetrahedral mesh, with Reck et al's approach [4], size limitation is about 200k cells with a 128MB graphics card. We propose to push this limit by storing the relevant information in textures in order to deal with grids made of several million cells.

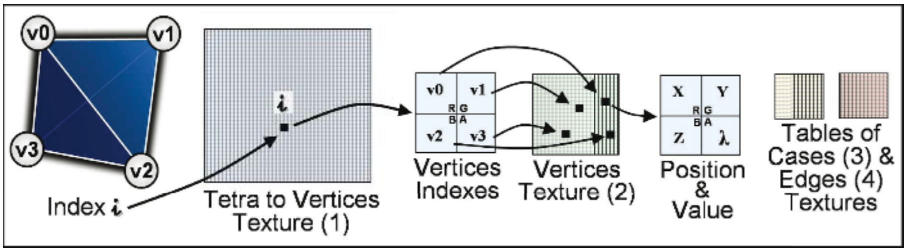


Fig. 2. Texture storage of the tetrahedra, the vertices and the description tables (each texel obviously contains four indexes)

Our goal is to store every needed data in a texture. Four indexes can be stored per texel of a texture, one in each RGBA component. As shown in Figure 2, tables of *edges* and *cases* are considered as 1D tables and are stored sequentially.

The following step of texture generation consists in iterating on each vertex of the grid to pack the corresponding data in a texture. The 3D geometry of the current vertex is packed in one texel, and so uses three of the four components of this texel. The remaining component is used to store the scalar value attached to the vertex. For rendering purposes, one more texel can be used to store a precomputed gradient.

Similarly, the last step iterates on each tetrahedron to pack the index of its four vertices in one texel. On the CPU side, each tetrahedron stores its unique entry in the corresponding texture.

Remark. Using our data storage method (Figure 2), the texture memory usage is evaluated to one texel per vertex without shading and one more texel per tetrahedron, plus few texels for the storage of the description tables. For a 128MB graphics card, a theoretical calculus shows that about 7 million tetrahedra could be stored on the graphics memory. This theoretical limit is higher than the practical limit because the graphics memory is not exclusively used for storing our textures. Nevertheless, current graphics cards commonly board 512MB (up to 1GB sometimes) of graphics memory. See results section 3 for more details.

2.2 Real-Time Rendering

Overview (Brute-force algorithm). After preprocessing, the isosurface extraction proceeds as follows:

- Done Once (Step 1)
  - Load the texture in the graphics memory
  - Load the vertex shading program
  - Set the isovalue
- For Each Cell (Step 2, see details below)
  - Send current cell index to the GPU

  - Send four vertices to the GPU, and implicitly execute the vertex shading program
- Updating (Step 3)
  - Update isovalue and go to Step 2

Sending Vertices to the GPU. The requested vertex number is sent to the vertex shading program through the position argument of the OpenGL vertex creation call:

```
glBegin(GL\_QUAD) ;
  glVertex2i(0, 0) ; // sends vertex 0
  glVertex2i(1, 1) ; // sends vertex 1
  glVertex2i(2, 2) ; // sends vertex 2
  glVertex2i(3, 3) ; // sends vertex 3
glEnd() ;
```

Vertex Shading Program. To code our vertex shading programs, we decided to use the high level programming language from Nvidia called CG[1], which is compatible with both ATI and Nvidia graphics cards. In this subsection, the pseudo CG code for extracting an isosurface from a tetrahedron is provided.

First, the algorithm reads the indexes of the vertices of the currently processed tetrahedron in the `tetra_texture` ((1) in Fig.2). From these indexes, the locations and values of the tetrahedron vertices are read in the `vertices_texture` ((2) in Fig.2):

```
float4 verticesIndex = tex1D ( tetra_texture, index_tetrahedron ) ;
float4 vertex_0 = tex1D ( vertices_texture, verticesIndex.r ) ;
float4 vertex_1 = tex1D ( vertices_texture, verticesIndex.g ) ;
float4 vertex_2 = tex1D ( vertices_texture, verticesIndex.b ) ;
float4 vertex_3 = tex1D ( vertices_texture, verticesIndex.a ) ;
float4 values = float4(vertex_0.a, vertex_1.a, vertex_2.a, vertex_3.a );
```

where tex1D is a function which performs a 1D texture lookup.

Then, the four vertices are assigned a 1 or a 0 flag depending on the isovalue, and the index in the table of *cases* is computed to determine the current configuration:

```
bool4 tested_vertices = ( values >= isovalue ) ;
int index = dot ( tested_vertices, float4(1,2,4,8) ) ;
// The symmetry of the table of cases is exploited
if ( index >= 8 ) { index = 15 - index } ;
```

According to the number of the vertex being processed (from 0 to 3, denoted `vertex_number`), the index of the intersected edge and of its end-points are retrieved from the `table_of_cases_texture` ((3) in Fig.2) and the `table_of_edges_texture` ((4) in Fig.2):

```
int intersected_edge = tex1D(table_of_cases_texture,index*4+vertex_number);
int vertex_index_0   = tex1D(table_of_edges_texture,intersected_edge*2   );
int vertex_index_1   = tex1D(table_of_edges_texture,intersected_edge*2+1 );
```

The vertex shading program then linearly interpolates the intersection of the edge with the isosurface, and finally moves the isosurface vertex to this location.

[1] http://developer.nvidia.com/object/cg_toolkit.html

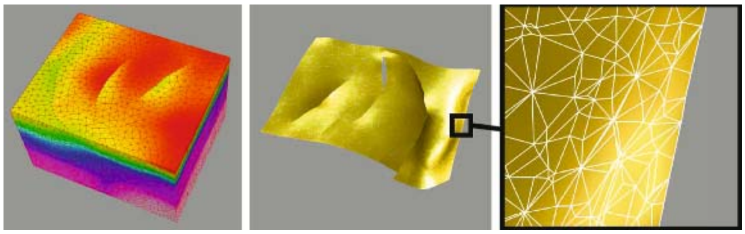


Fig. 3. Isosurface rendering of the fluid pressure field in a tetrahedralized geological model

Comments. Texture lookup in vertex shader unit is, at this time, relatively slow. Nvidia's documentation[2] indicates that a Geforce 6800 is theoretically capable of processing more than 600 million vertices per second. Add to this bench a vertex texture lookup for each vertex, and the performance falls to 33 million processed vertices per second. This drop of performance is due to long latencies introduced when a texture lookup is requested. 'Latency' means that vertex shading program could execute some assembly codes which are not related to the texture lookup (even other texture lookups) while waiting for this lookup. Therefore, grouping the texture lookups in the vertex shading program parallelizes texture lookup latencies and optimizes performance.

3 Results and Comparisons

Our algorithm has been tested on several tetrahedral grids used in geological modeling, made of up to 5 million cells (see for instance Fig.3). All benchmarks (Fig.4) are done on a laptop with an Intel Centrino 2Ghz processor with 1GB of RAM and an Nvidia QuadroFX Go 1400 256MB PCI-Express graphics card (6th generation from this manufacturer). This card has only 3 vertex shading units, against 8 units in a 7th generation card: using the last 7th generation from Nvidia would have further increased the differences between the CPU and the GPU methods. Notice that the desktop equivalent to our testing graphics card is now a low cost hardware.

Figure 4 presents the number of tessellated tetrahedra per second in a brute-force algorithm with shading for several grid sizes using different methods: a pure CPU based, a GPU register based [2,4] and our GPU texture based extraction algorithm. The GPU register curve can be split into three parts: below 600K tetrahedra, the whole grid fits in graphics memory, figuring a constant processing speed of about 6.7 million tetrahedra per second; between 600k tetrahedra and about 1 million, the grid does not fit in graphics memory and the PCI-Express bus is used to swap some data with the RAM, resulting in an important performance drop; for more than one million tetrahedra, both the 256MB of graphics memory and the 1GB RAM are fully loaded, and then performance

[2] http://developer.nvidia.com/object/using_vertex_textures.html

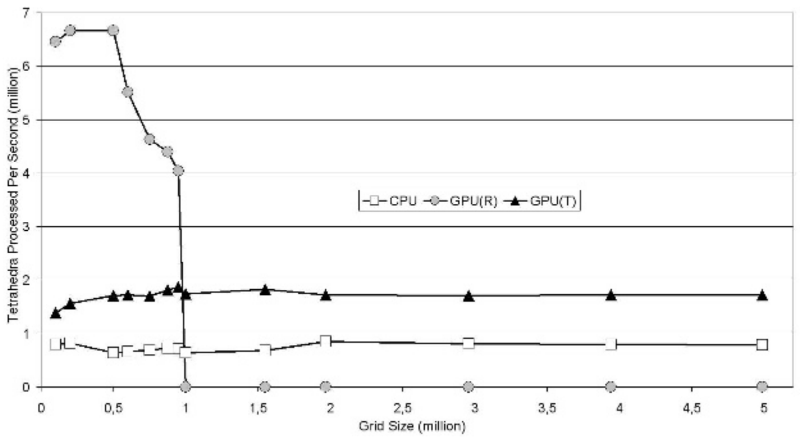


Fig. 4. Number of tetrahedra tessellated per second versus the size of the grid in brute force mode with shading. GPU(R) denotes GPU method using Registers [2,4] and GPU(T) our method using Textures.

drops tremendously, since swapping between hard-drive and PC memory is necessary. Pascucci and Reck et al. [2,4] note that performance drops by a factor of 10 to 20 with an AGP graphics card when the whole data do not fit in video memory. They also suggest that using a PCI-Express card may greatly help, which is confirmed by this work. In our tests, at the limit of a grid counting 1 million tetrahedra, the speed is still near 4 million tetrahedra processed per second (to compare with about 0.5 million processed per second using an AGP card). Both CPU and GPU texture-based methods are linear, and our texture-based method applied to tetrahedra appears to be, on average, twice faster. It is, hence, slower than the GPU register-based method but only for grids made of less than 1 million tetrahedra. Above (up to at least 5 million tetrahedra), our method remains linear. Thanks to these benchmarks, choosing the fastest method for isosurface extraction can be done automatically depending on the grid size and the available memory (RAM and graphics memory).

Comments. The results above were obtained using a brute-force grid traversal to prove the efficiency of the compared algorithms. These methods, including ours, can be combined with algorithms that narrow marching time by previously discarding non-intersected cells (section 1.2). In this case, the whole texture is uploaded once to the graphics memory and only the list of indexes corresponding to intersected cells is sent to the graphics card at the rendering stage. For example, with our approach combined with an Interval Tree [8], we get an acceleration factor of about 4 times on average.

Latencies in accessing textures in the vertex shading unit will soon be improved, according to manufacturers' plans for their graphics cards supporting Shader Models 3.0 and 4.0. These accesses would be as fast as accessing a texture in the pixel shading unit, so about as fast as accessing a register. Then, our method would combine the speed of the register based approaches [2,4] and the advantages of the texture based storage. Also, most manufacturers[3] are considering the possibility of unifying the vertex and pixel shading units under only one unique hardware unit. Since the pixel shading unit is much more powerful than the vertex shading unit, our method would be strongly accelerated by this evolution. Moreover, these manufacturers will introduce the Geometry Shading Unit in their next generation of graphics cards, enabling the creation of vertices within the GPU. This functionality will limit the redundancy of the computation of the configuration index, and will speed up the extraction process.

4 Conclusion

Our method introduces a hardware-accelerated algorithm to efficiently tessellate very large unstructured tetrahedral meshes, as used for finite element modeling. This method overcomes several limitations by using textures to store efficiently the whole data without introducing redundancies through shared vertices, and limits the AGP/PCI-Express transfers. Our technique handles grids up to five million tetrahedra at least, while improving tessellation performance as compared to CPU extraction. Moreover, it supports both brute-force extraction and extraction after discarding non-intersected cells using, e.g., an Interval tree, an Octree, a Seed Set, etc...

Future works include studying the extraction of isosurface on strongly heterogeneous grids using GPU acceleration; the first step in this direction will be the support for hexahedral meshes. Moreover, non-linear interpolation in isopolygons could be interesting to improve rendering quality. This study could also be extended to time-varying data, volume-rendering and parallel processing.

Acknowledgements

The authors thank the members of the GOCAD research consortium for their support(`www.gocad.org`), especially EarthDecision for providing the Gocad software. The authors also thank T. Frank and L. Macé for their help.

References

1. Kipfer, P., Westermann, R.: GPU construction and transparent rendering of isosurfaces. In Greiner, G., Hornegger, J., Niemann, H., Stamminger, M., eds.: Proc Vision, Modeling and Visualization 2005, IOS Press, infix (2005) 241–248
2. Pascucci, V.: Isosurface Computation Made Simple: Hardware Acceleration, Adaptive Refinement and Tetrahedral Stripping. In: IEEE TVCG Symposium on Visualization '04. (2004) 293–300

[3] http://downloads.gamedev.net/features/programming/atid3d10/ATI-TheFutureof PCGaming.pdf

3. Klein, T., Stegmaier, S., Ertl, T.: Hardware-accelerated reconstruction of polygonal isosurface representations on unstructured grids. In: Proc. 12th Pacific Conference on Computer Graphics and Applications (PG'04), Washington, DC, USA, IEEE Computer Society (2004) 186–195
4. Reck, F., Dachsbacher, C., Grosso, R., Greiner, G., Stamminger, M.: Realtime isosurface extraction with graphics hardware. In: Proc. Eurographics. (2004)
5. Bajaj, C.L., Pascucci, V., Schikore, D.R.: Fast isocontouring for improved interactivity. Proc. Symposium on Volume Visualization (1996)
6. van Kreveld, M.: Efficient methods for isoline extraction from a tin. GIS, 10:523-540 (1996)
7. van Kreveld, M., van Oostrum, R., Bajaj, C.L., Schikore, D.R., Pascucci, V.: Contour trees and small seed sets for isosurface traversal. Proc. Thirteenth ACM Symposium on Computational Geometry (1997)
8. McCreight, E.M.: Priority search trees. SIAM J. Comput. 14, 257-276 (1985)
9. Wilhelms, J., Gelder, A.V.: Octrees for faster isosurface generation. ACM Trans. Graph. **11** (1992) 201–227
10. Silva, C.T., Mitchell, J.S.B.: The lazy sweep ray casting algorithm for rendering irregular grids. IEEE Transactions on Visualization and Computer Graphics **3** (1997) 142–157
11. Livnat, Y., Shen, H.W., Johnson, C.R.: A near optimal isosurface extraction algorithm using the span space. IEEE Transactions on Visualization and Computer Graphics **2** (1996) 73–84
12. Cignoni, P., Marino, P., Montani, C., Puppo, E., Scopigno, R.: Speeding up isosurface extraction using interval trees. IEEE Transactions on Visualization and Computer Graphics **3** (1997) 158–170
13. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. ACM SIGGRAPH'87 (1987)
14. Shirley, P., Tuchman, A.A.: A polygonal approximation to direct scalar volume rendering. Proc. San Diego Workshop on Volume Visualization, Computer Graphics **24** (1990) 63–70
15. Bloomenthal, J.: Polygonization of implicit surfaces. Computer Aided Geometric Design **5** (1988) 53–60
16. Lévy, B., Caumon, G., Conreaux, S., Cavin, X.: Circular incident edge lists: a data structure for rendering complex unstructured grids. In Ertl, T., Joy, K., Varshney, A., eds.: Proc. IEEE Visualization. (2001)
17. Caumon, G., Lévy, B., Castanié, L., Paul, J.C.: Visualization of grids conforming to geological structures: a topological approach. Computers and Geosciences **31** (2005) 671–680
18. Engel, K., Kraus, M., Ertl, T.: High-Quality Pre-Integrated Volume Rendering Using Hardware-Accelerated Pixel Shading. In: Eurographics / SIGGRAPH Workshop on Graphics Hardware '01. Annual Conference Series, Addison-Wesley Publishing Company, Inc. (2001) 9–16
19. Castanié, L., Lévy, B., Bosquet, F.: VolumeExplorer: Roaming large volumes to couple visualization and data processing for oil and gas exploration. In: IEEE Visualization. (2005) 32

Enhancing Information on Large Scenes by Mixing Renderings

Vincent Boyer and Dominique Sobczyk

L.I.A.S.D. - Université Paris 8
2 rue de la liberté
93526 Saint-Denis Cedex - France
{boyer, dom}@ai.univ-paris8.fr

Abstract. We propose a new model for visualization of high scale scenes. It is designed to enhance pertinent informations that become quickly viewable on a large scene. It consists in mixing different kind of rendering techniques in the same frame. This method is achieved in real-time during the rendering process using GPU programming. Moreover rendering techniques used and key points defined by the user can be interactively changed. We present our model, and a new non-photorealistic rendering techniques. Images produced look better and provide more informations than traditional rendering techniques.

1 Introduction

This paper presents a new method to render high scale 3D models. The aims of this model are both a better looking and a more informative image. Rendering is the process of generating an image from a given model. This can be achieved using software programs (generally through a graphic programing library) or/and GPU programs. Generally renderings are classified in two main categories: photo-realistic and by opposition non-photorealistic.

Photo-realistic renderings of a high scale scene do not allow to enrich the image produced. By opposition non-photorealistic renderings try to enhance the transmission of information in a picture [1], [2]. Based on the idea that a new rendering will help the user to visualize 3D models, a lot of previous techniques have been developed. We present some of these ones organized in different topics:

- deformation of the 3D mesh according to the viewpoint. Rademacher [3] has proposed to create a view dependent model. It consists of a base model and a complete description of the model shape from key viewpoints. This can be used especially by cartoonists. A generalization of view-dependent deformations was presented by Martin et al. [4]. A control function to relate position and transformation is used. Other works have been presented and for example Wood et al. [5] proposed the generation of panoramas for a given 3D scene and a camera path.
- edges extraction. A lot of well-known works have been done to extract edge features like silhouettes, boundaries and creases from 3D models [6], [7], [8],

[9]. Frontfacing and backfacing can be used to detect silhouette and normal maps is one of the real-time methods to find creases and boundaries. Generally, a preprocessing operation should be done to convert a 3D model in another data structure in order to obtain edges more easily (for example half-edge data structure maintains edge adjacency). The main applications of these techniques are architectural and technical illustrations.
– artistic shading. Hatching [10] [11], cartoon shading [12] and celshading are examples of non-photorealistic shading. They replace Gouraud or Phong shading to convey three-dimensional structure of objects in an image. Hatching is based on the curvature model while cartoon and celshading choose the fragment color in a one-dimensional texture using the dot product between light direction and the normal. Note that celshading often includes outline shading.

As one can see, there is not one solution to enhance the transmission of informations and each technique has its favorite applications, advantages and drawbacks.

The motivation of this paper can be summarized in one question: why do we use one and only one rendering to produce an image? This paper presents a model to combine different renderings in the same frame. Each rendering and its influence on the scene will be chosen by the user and can be changed interactively.

In the following, we present the general algorithm of our model and detail the solutions to solve each problem. Then images produced by our system are detailed and in conclusion the limitations and future works are presented.

2 The Model

Our model mixes different renderings at the same time. The user should specify which and where renderings will be used. The GPU programming can be achieved in vertex or/and fragment shaders. The vertex shader is used to precompute values needed in the fragment shader. The main process of our model is realized on fragments. For a given fragment, different renderings can be applied simultaneously and should be blended. Due to the real-time constraint, the computation should be done on GPU.

The specifications given by the user are:

1. **key points** which are points of the 3D space. In the following n is the number of key points given by the user;
2. renderings used and for each of them four distances dr_0, dr_1, dr_2 and dr_3 where $[dr_0, dr_3]$ is the range within the rendering is used.
   - dr_0: the minimal distance in order to use this rendering;
   - dr_1: the minimal distance where we maximize the use of this rendering. Between dr_0 and dr_1 the rendering will be shaded;
   - dr_2: the maximal distance where we maximize the use of this rendering;
   - dr_3: the maximal distance in order to use this rendering. Between dr_2 and dr_3 the rendering will be shaded.

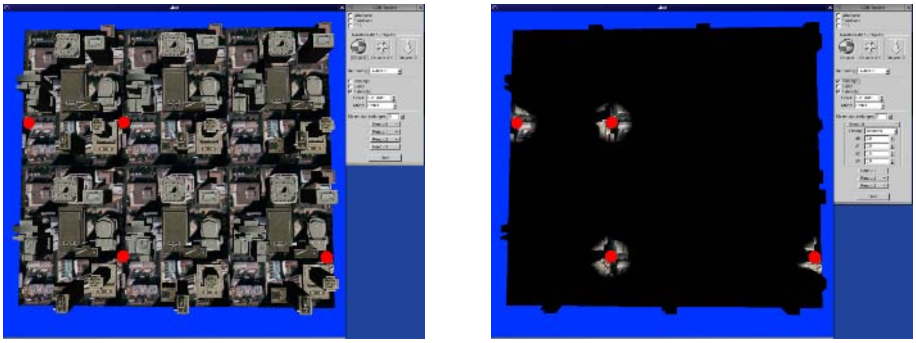


Fig. 1. Example of key points and distance given by the user

Left part of figure 1 presents four key points given by the user and their visualization on the scene and the right part shows the distances for a texture rendering given by the user on the interface and the result on the scene. The distances dr_0 and dr_1 are equal to 0.0, dr_2 is equal to 0.5 and dr_3 is equal to 0.8. Thus texture is applied in the range of 0.0 to 0.5 from the key points and in the range between 0.5 to 0.8 the texture is applied shaded with black. In this example we gave four key points, one rendering and its associated distances. Note that the number of key points, the coordinates of key points, the number of rendering used, the renderings chosen and the distances used could be modified dynamically by the user.

The general form of our algorithm is:

```
For a given fragment $F$
    Compute the closest distance $d$ to the key points ①
    For each rendering
        Compare the distances given by the user and distance $d$ and compute
        a weighted value $v$ ②
        Compute the color and weight it by $v$ ③
    Sum the colors previously obtained and clamp it between 0 and 1 for
    each component.
```

In the following we detail the methods used to compute the distance to the key points ①, the ratio used for each rendering ② and present the computation of the fragment color using a rendering ③. We also present a new rendering. These computations are realized only in the GPU and we obtain real-time renderings. Remark that at the present time, fragment shaders do no permit to make loops on data which do not reference a texture. So even if we present the algorithms including loops for the reader, we unfold it. This limit should disappear with future versions of graphic cards and shaders.

2.1 Distance to Key Points

The user defines key point(s). This section explain how to compute the distance d. We propose three modes to consider key points:

- single points: the distance d is the minimal euclidean distance between fragment and each key point(s);
- points of a polyline: the key points form a polyline and d is the minimal distance between fragment and this polyline. The algorithm used to compute d for a fragment F is:

```
For each segment $P_iP_{i+1}$ ($i \in [0; n-2]$) of the polyline
    If $\boldsymbol{P_iP_{i+1}} \odot \boldsymbol{P_iF} < 0$ then $d_i$ is the euclidean distance between $P_i$ and $F$
    Else if $\boldsymbol{P_iP_{i+1}} \odot \boldsymbol{P_iF} > \|\boldsymbol{P_iP_{i+1}}\|^2$ then $d_i$ is the euclidean distance between $P_{i+1}$ and $F$
    Else $d_i = \frac{\|\boldsymbol{P_iP_{i+1}} \otimes \boldsymbol{P_iF}\|}{\|\boldsymbol{P_iP_{i+1}}\|}$
$d = \min d_i$
```

  where $\odot$ represents the dot product of two vectors, $\otimes$ the cross product of two vectors and $\|\boldsymbol{P_iP_{i+1}}\|$ the norm of the vector $\boldsymbol{P_iP_{i+1}}$. Note that $\|\boldsymbol{P_iP_{i+1}}\|^2$ is equal to $\boldsymbol{P_iP_{i+1}} \odot \boldsymbol{P_iP_{i+1}}$ and is computed using the dot function on the fragment shader;
- points of a polyline loop: in this case, we consider the n key points as n points of a polyline but $P_{n-1}P_0$ is considered as a polyline segment. The distance d is computed using the algorithm described for the polyline mode including one more segment (i.e. $P_{n-1}P_0$).

Figure 2 shows the influence of the mode used in the computation of the distance d. The polyline mode is presented at the left and the polyline loop mode is shown on the right while the single points mode is presented on the left of figure 1.

2.2 Compute the Weighted Value for Each Rendering

The user enters his choices for renderings used and for each of them four associated distances (dr_0, dr_1, dr_2, dr_3). We have previously computed the distance d between the fragment and the key points. For each rendering used, we compute a coefficient v. v is then applied to weight the rendering. The sum of weighted rendering on a fragment allows us to mix renderings on it. The algorithm is:

```
For each rendering
    If ($d \in [dr_0; dr_3]$) then
        If ($d < dr_1$) $v = \frac{d - dr_0}{dr_1 - dr_0}$
        Else if ($d \leq dr_2$) $v = 1.0$
        Else $v = \frac{dr_3 - d}{dr_3 - dr_2}$
    Else $v = 0$
```

Remark that for a given fragment f, it is possible to have zero, one, two and more renderings. Each rendering is weighted independently and v can be viewed as the alpha channel for a rendering on a fragment. Finally the fragment color is clamped between 0 and 1. We include the ability to blend the image with the background.

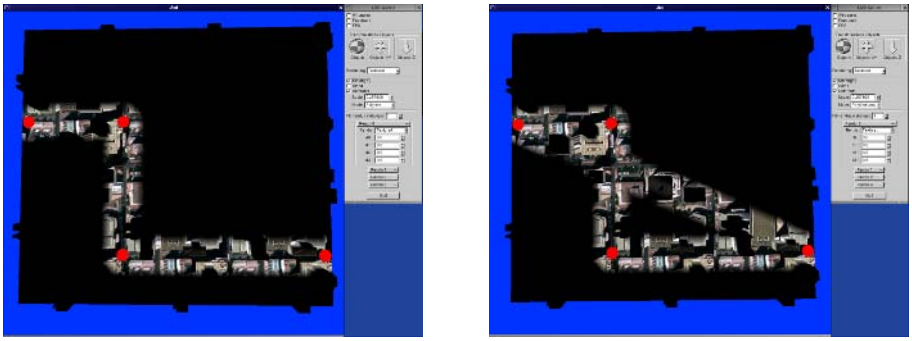


Fig. 2. Example of polyline and polyline loop use

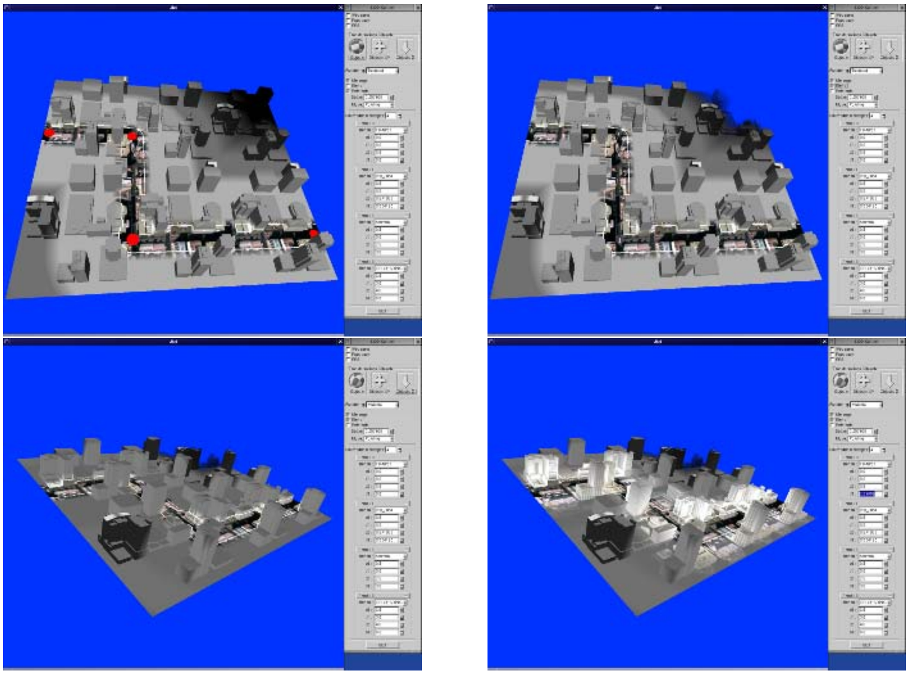


Fig. 3. Compute the coefficients and render the scene

Figure 3 illustrates this part: the top right screenshot shows the use of blending; on the bottom image, the distance dr_3 of texture rendering has been changed and fragments are rendered with texture, wb_dist (see next section) and material when $d = 0.55$.

2.3 Rendering

Texture rendering, material rendering, toon shading and celshading (using texture and outline) have been implemented in GPU fragment shader. Vertex shader is used to transform vertex coordinates and to compute necessary data in the fragment shader. These shaders are well-known and many books describe it, see for example[13].

We implement a new rendering, named **wb_dist** for "white to black distance". This rendering is very simple and is based on the coefficient previously computed. The fragment color components received the coefficient v. Then, if the fragment is in $[dr_1; dr_2]$, the color is white and alpha is 1, else the fragment color is an achromatic color and alpha is in$[0; 1[$ depending on the distance d. For example, it allows us to make a halo effect when dr_1 is closest to dr_2. This can be used to mark a key point or a path as shown in figure 4. Also, other effects like NASA aerogel can be easily produced.

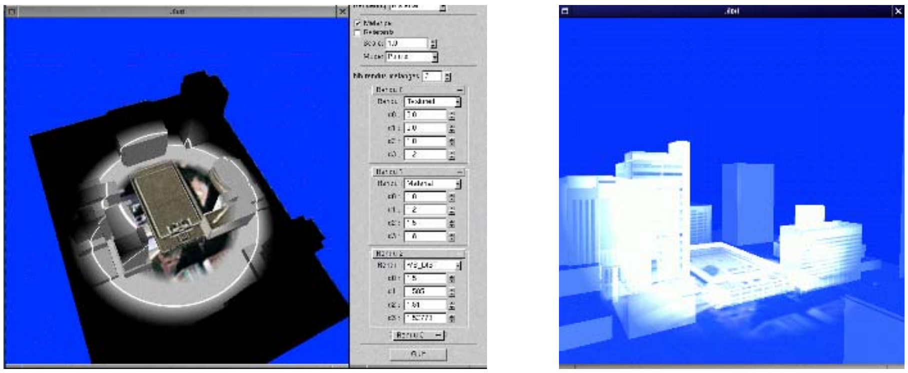


Fig. 4. Halo and "NASA aerogel" rendering

3 Images and Results

This section presents images produced with our model. Those have been produced on a PC pentium IV 3.6Ghz with 1Go of memory and a nvidia graphic card Quadro FX 1400. The top of figure 5 presents a first part of the path described with key-points. As one can see, the halo rendering and the mix of renderings help the user to obtain the essential information needed. The bottom of the figure presents two other views where the used renderings have been changed. Indeed, in the first of these, textured rendering then material rendering are used according to the distance of the polyline and in the second one this is the opposite (i.e. material then textured rendering). The minimal frame rate obtained for this scene composed by 20 000 triangles is 27 frames per second with 4 renderings (textured, material, wd_dist and celshading) used simultaneously and 4 key-points. In fact, the

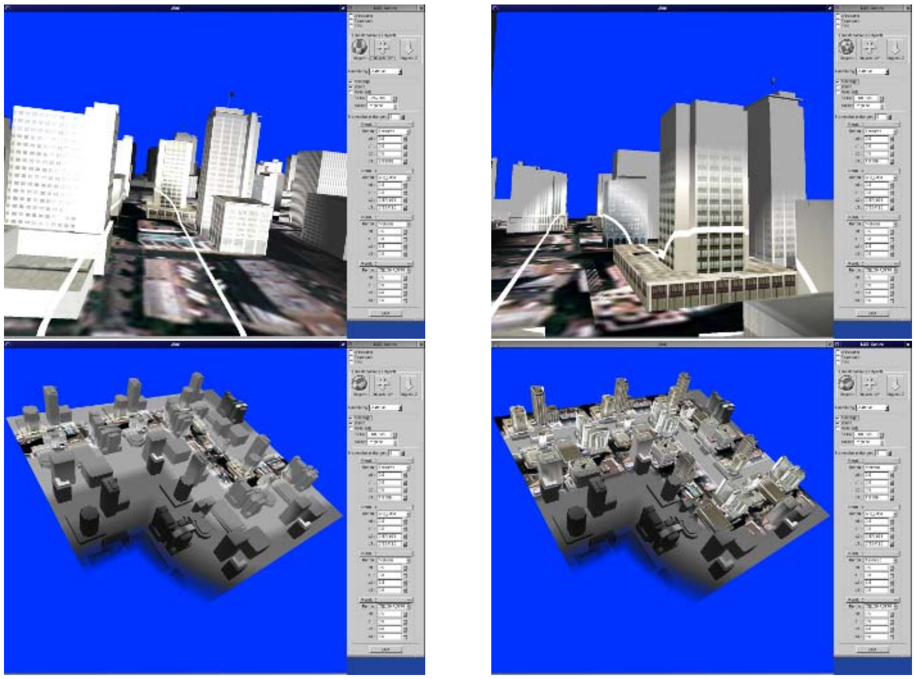


Fig. 5. Other views of San Diego

number of key-points used has no influence on the frame rate (i.e time computation needed for the distance computation is negligible).

4 Conclusion

We have proposed the first model to mix renderings. Based on GPU programing, this model is real time for large scale scenes visualization. A collection of various rendering shaders have been implemented one of them being a new one and other ones can be easily added. The user can apply particular renderings on each part of the scene. This permits to have only advantages of renderings and never their drawbacks. Moreover every parameters can be modified by the user dynamically and it is very intuitive. This model allows us to focus on a point or a path in a large scene. We can apply immediately, as shown in examples, this model to a car journey. Other applications like for example to show influence of a construction on an environment, visualization of the same object in a place (fire extinguisher in a building, electric cables in an oil rig, . . .), or educational courses can be realized.

Future works will be done to propose other interpolations in order to compute the coefficients (non-linear interpolations) and to automatically import other renderings.

References

1. Strothotte, T., Schlechtweg, S.: Non-photorealistic computer graphics: modeling, rendering, and animation. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2002)
2. Gooch, B., Gooch, A.: Non-Photorealistic Rendering. A K Peters (2001)
3. Rademacher, P.: View-dependent geometry. Proceedings of SIGGRAPH 99 (August 1999) 439–446 ISBN 0-20148-560-5. Held in Los Angeles, California.
4. Martin, D., Garcia, S., Torres, J.C.: Observer dependent deformations in illustration. In: Non-Photorealistic Animation and Rendering 2000 (NPAR '00), Annecy, France (June 5-7,2000)
5. Wood, D.N., Finkelstein, A., Hughes, J.F., Thayer, C.E., Salesin, D.H.: Multiperspective panoramas for cel animation. Proceedings of SIGGRAPH 97 (August 1997) 243–250 ISBN 0-89791-896-7. Held in Los Angeles, California.
6. Markosian, L., Kowalski, M.A., Trychin, S.J., Bourdev, L.D., Goldstein, D., Hughes, J.F.: Real-time nonphotorealistic rendering. Proceedings of SIGGRAPH 97 (August 1997) 415–420 ISBN 0-89791-896-7. Held in Los Angeles, California.
7. Gooch, A., Gooch, B., Shirley, P., Cohen, E.: A non-photorealistic lighting model for automatic technical illustration, SIGGRAPH 98 (1998)
8. Interrante, V., Fuchs, H., Pizer, S.: Illustrating transparent surfaces with curvature-directed strokes. IEEE Visualization '96 (1996) 211–218 ISBN 0-89791-864-9.
9. Sousa, M.C., Prusinkiewicz, P.: A few good lines: Suggestive drawing of 3d models. Computer Graphics Forum **22** (2003)
10. Winkenbach, G., Salesin, D.: Rendering parametric surfaces in pen and ink. Proceedings of SIGGRAPH 96 (August 1996) 469–476
11. Hertzmann, A., Zorin, D.: Illustrating smooth surfaces. Proceedings of SIGGRAPH 2000 (July 2000) Held in New Orleans, Louisianna.
12. Lake, A., Marshall, C., Harris, M., Blackstein, M.: Stylized rendering techniques for scalable real-time 3d animation. In: Non-Photorealistic Animation and Rendering 2000 (NPAR '00), Annecy, France (June 5-7,2000)
13. Rost, R.: OpenGL Shading Language. Pearson (2004)

Auto-focusing in Extreme Zoom Surveillance: A System Approach with Application to Faces

Yi Yao, Besma Abidi, Michael Tousek, and Mongi Abidi

The University of Tennessee, Knoxville, TN, 37996

Abstract. Auto-focusing is an indispensable function for imaging systems used in surveillance and object tracking. In this paper, we conduct a study of an image-based passive auto-focusing control for high magnification (>50×) systems using off-the-shelf telescopes and digital camcorders with applications to long range near-ground surveillance and face tracking. Considering both speed of convergence and robustness to image degradations induced by high system magnifications and long observation distances, we introduce an auto-focusing mechanism suitable for such applications, including hardware design and algorithm development. We focus on the derivation of the transition criteria following maximum likelihood (ML) estimation for the selection of adaptive step sizes and the use of sharpness measures for the proper evaluation of high magnification images. The efficiency of the proposed system is demonstrated in real-time auto-focusing and tracking of faces from distances of 50m~300m.

1 Introduction

PTZ cameras are commonly used in indoor and outdoor surveillance systems [1, 2]. Most of these cameras provide a maximum optical magnification in the range of 20×~25×. However, for long range surveillance and target identification, this magnification is insufficient. To extend the optical zoom capability beyond 50×, we exploited digital imaging systems with scopes (telescopes and spotting scopes) in near-ground surveillance [3].

As the first step in building a high magnification imaging system using off-the-shelf equipment, we studied several setups based on various scopes, eyepieces, and digital cameras/camcorders [3]. The chosen combination of a Celestron telescope (GPS 11) and a Sony camcorder (TRV730) is able to achieve a system magnification of up to 1800×, which is sufficient for observing human faces from a distance of 1km.

For this system to be useful in real-time tracking scenarios, it is critical to keep the moving target in focus. In a composite imaging system, the focus of the scope plays a dominant role. Although digital cameras are equipped with auto-focusing function, scopes are available only with manual focus control. To facilitate complete remote and automatic control of this high magnification imaging system, the auto-focusing capability needs to be integrated.

However, auto-focusing for composite imaging systems with high magnifications is a newly emerged research topic and not well addressed in literature. Moreover, the application of existing auto-focusing algorithms to such systems is non-trivial. Fig. 1 depicts the typical responses of the conventional Laplacian sharpness measure [4] for low (2.28×) and high (245× and 1500×) magnification sequences collected at uniformly sampled camera focus positions. Compared with low magnification imaging systems, we experience two major difficulties. (1) For a large visible distance, our high magnification imaging system involves a significantly wider dynamic focus range varying from 20m up to 1000m (infinity). (2) The collected images suffer substantially from degradations such as increased image noise level and severe image blur from high magnification and air turbulence, producing time varying and noisy sharpness measures. These two constraints impose additional requirements, especially the speed of convergence and robustness to image degradations, on the design of suitable auto-focusing algorithms.

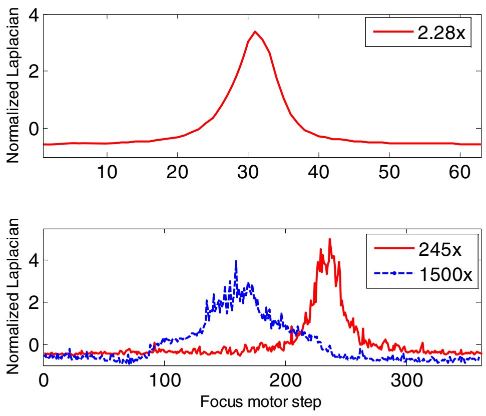


Fig. 1. Typical responses of sharpness measures for low (2.28×) and high (245× and 1500×) magnification sequences. The dynamic focus range of a low magnification imaging system is typically much smaller compared with that of a high magnification imaging system. The sharpness values of high magnification images are severely corrupted by degradations such as increased image noise level and blur.

In consideration of these difficulties, a sequential search with variable step size is selected as the backbone search strategy. The sequential search completes peak detection in one sweep, nearly eliminates changes in motion direction, and saves on motor steps. A variable step size optimizes the motor step distribution in the sampled focus range and minimizes the number of iterations. To improve the algorithm's speed of convergence and robustness to image degradations, we focus on the derivation of the transition criteria and the selection of sharpness measures. The transition criteria are obtained from ML estimation under the assumption of Gaussian distribution and from statistical studies of the collected image sequences with various magnifications and scene structures. Meanwhile, two types of sharpness measures are employed in the ramp and peak regions according to the way they respond to out-of-focus blur. The summation of two types of sharpness measures can also be used for a reduced noise level.

The remainder of this paper is organized as follows. Section 2 reviews existing image-based passive auto-focusing algorithms and compares their performances. Section 3 briefly describes our high magnification imaging system. In section 4, an auto-focusing algorithm for high magnification imaging systems is proposed and its efficiency is validated via both offline and real-time image sequences in section 5. Section 6 concludes this paper.

2 Literature Review

Among the existing auto-focusing methods, we are interested in image-based passive auto-focusing approaches primarily because of their simple configuration in hardware. Within this category, one major area is the estimation of depth by analyzing the degree of focus in a sequence of images [5]. In our application a considerable amount of blur comes from high magnification rather than improper focus. The simple relation between blur and depth is not entirely valid. For this reason, the use of this type of methods in high magnification systems remains questionable.

In the second main branch of image-based passive auto-focusing algorithms, optimum focus is found by searching for the focus location which yields an image with the highest sharpness measure. Various search strategies have been developed. The Fibonacci search is the best-known algorithm [4], which guarantees that the maximum of the criterion function is found within a known number of iterations depending only on the dynamic focus range. The hill-climbing search divides the procedure into two stages: out-of-focus region (coarse) search and focused region (fine) search. Given a heuristic choice of step magnitudes, the hill-climbing search is able to converge to the optimal focus. A number of hill-climbing algorithms have been proposed with modifications regarding step size selection, termination criteria, search window, *etc* [6, 7].

Variations are introduced to these basic algorithms for a better performance. In the fine search stage, the image sharpness is evaluated at three focus locations and the samples are fitted to a quadratic or a Gaussian function, the maximum of which is the estimated focused position [8]. To avoid the back and forth motor motion required by the Fibonacci search, Kehtarnavaz *et al.* proposed a sequential search algorithm, referred to as the rule-based search (RS), where the step size is varied according to the distance from the best focus location [9].

A detailed review and comparison of relevant search algorithms can be found in [10]. Their performances in conjunction with different sharpness measures are examined using both low and high magnification image sequences based on three criteria: accuracy, speed of convergence described by the number of iterations and the number of motor steps traveled before the optimal focus is obtained, and robustness to image degradations and parameter selection. In general, the hill-climbing search is sensitive to parameter selection. With the Fibonacci search, the number of iterations for a given focus range is usually fixed. However, the Fibonacci search involves the most back-and-forth motion and therefore the most motor steps. The RS algorithm involves only unidirectional movements and hence requires fewer motor steps. The use of function approximation avoids unnecessary iterations during the fine search stage, thereby reducing the total number of iterations and motor steps. Overall, the RS and the Fibonacci search with function fitting (FF) outperform other tested algorithms.

3 System Description

Our high magnification imaging system, equipped with high speed and automatic pan/tilt and focus control abilities, is shown in Fig. 2. To fully explore the optical capabilities of both the Celestron lens and the Sony camcorder, an afocal coupling is selected. The Celestron lens is connected to the Sony camcorder via a Celestron Plössl 40mm eyepiece or a Meade 26mm eyepiece. The focal length of the Celestron lens is 2800mm and the Sony camcorder has a 47mm~846mm zoom capability. The achievable system magnification is approximately 70× to 1800×.

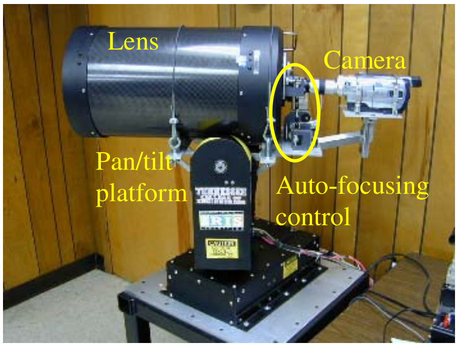


Fig. 2. System hardware setup: Fully motorized pan/tilt/zoom control and auto-focusing capability facilitate remote and automatic control. The resulting system can perform object tracking and monitoring in the same fashion as commercial PTZ cameras.

The Celestron lens's existing focus control features a manually operated control knob requiring 40 full turns to cover the complete focus range. To automate it, we coupled the control to an Animatics SmartMotor through a gear drive of our own design. The main requirement was that the system be precise enough to give repeatable control positioning with increments as fine as the smallest resolution which starts to produce noticeable degradation in the resulting images. The empirical minimum resolution is less than 40 degrees of knob rotation. When converted to motor steps and normalized to the minimum resolution, the dynamic range is -200 to 200 steps.

4 Auto-focusing with High Magnification

In light of the system limitations – *i.e.* wide focus dynamic range and noisy sharpness measures – and the performance comparisons among various search algorithms [10], sequential search algorithms with variable step sizes appear to be the most promising techniques. The remaining questions are: (1) when and how to change the step size and (2) how to evaluate image sharpness accurately. The derivation of the transition criteria and the selection of sharpness measures answer the above questions, respectively.

Compared with other sequential search algorithms, the proposed transition criteria are designed based on ML estimation and have the potential to employ maximum *a posteriori* (MAP) estimation. The advantage becomes evident for high magnification imaging systems, where the resulting sharpness measures are considerably noisy. Simple thresholding is insufficient and leads to false detection or detection failure (no peak is detected). The high system noise level justifies and calls for the use of ML or MAP estimation.

A typical sharpness measure curve can be divided into three regions: peak, ramp, and saturation. For a rapid convergence, a gradual ramp region and a sharp peak region are preferred. Based on their responses to camera focus, two types of sharpness measures are recommended. Autocorrelation and gradient based measures are well suited for the search in the ramp and peak regions, respectively. The use of different sharpness measures alternates in the same fashion as the step size, depending on the current search region.

4.1 Transition Criteria Derivation

The step sizes are adjusted adaptively throughout the search process according to the current focus location. Small, medium, and large step sizes are used in the peak, ramp, and saturation regions, respectively. From the viewpoint of a state transition machine (STM), three distinctive states can be defined accordingly. The state transition representation associates the search process with an estimation process, where an optimal sequence of state transitions is retrieved given a sequence of noisy observations and a pre-defined structure (states and transition hypothesis). Consequently, MAP and ML estimation can be applied. Most of the sequential search algorithms use empirical thresholds to govern the step size transitions. Based on the STM representation, these thresholds can be indeed derived from ML estimation.

To build probabilistic models for state transitions, the statistical behavior of the sharpness measures is studied. The search process is divided into two stages: the pre-peak stage where no peak is detected and the post-peak stage where a possible peak is detected. In the pre-peak stage, the determinant variable is ΔS, the difference between consecutive sharpness measures, while in the post-peak stage, the focus is shifted to S. In our implementation, $S_{\max}$, the recorded maximum sharpness measure, is used as a reference. We examine the statistical behavior of $\Delta S / S_{\max}$ and $S/S_{\max}$ and

Table 1. Transition criteria. Assuming that the current state is *peak* and $\Delta S < 0$, C_{down} increases by one. The successive state is *ramp* if $C_{down} \geq C_{th}$ and remains in *peak* otherwise. From our experiments, C_{th}=3 is recommanded.

		End state		
		Peak	*Ramp*	*Saturation*
Start state	*Peak*	$\Delta S < 0,\ C_{down}{+}{+}$		$S \leq 0.24 S_{max}$
		$C_{down} < C_{th}$	$C_{down} \geq C_{th}$	
	Ramp	$\Delta S > 0.23 S_{max}$	$\Delta S > 0.09 S_{max}$, $C_{flat}{+}{+}$	
			$C_{flat} < C_{th}$	$S \leq 0.24 S_{max}$ or $C_{flat} \geq C_{th}$
	Saturation	$\Delta S > 0.23 S_{max}$	$\Delta S > 0.09 S_{max}$	Otherwise

obtain the thresholds assuming that both variables obey a Gaussian distribution. In practice, to avoid back-and-forth switches caused by noise, some state transitions are issued only when the corresponding transition criteria are satisfied C_{th} times. The threshold C_{th} is obtained empirically. The following counters, C_{down} and C_{flat}, are defined for the ramp region in the post-peak stage and the saturation region, respectively. Table 1 summarizes the major transition criteria.

4.2 Sharpness Measure Selection

The proper use of sharpness measures is also of great importance to the overall system performance. Different sharpness measures respond to the changes in camera focus in quite different ways. Variance based sharpness measures produce gradual slopes while gradient based sharpness measures produce sharp peaks [11]. However, the performance of variance based sharpness measures deteriorates for high magnification images. In some cases, they could not even preserve the desired unimodal shape of an appropriate sharpness measure [10].

From the analysis of their properties [10], we observe that autocorrelation based measures (ACF) [12] generate responses with varying slopes depending on the window sizes used, as shown in Fig. 3(a). Measures with large window sizes produce wide peaks and gradual slopes, which can be used for the coarse search stages (saturation and ramp regions). Gradient based measures are used in the fine search stage (peak region).

In practice, the combination of two types of sharpness measures can be used to improve the response shape and to suppress noise. The summation of the Tenengrad [4] and ACF with a widow size of 10 (ACF_{10}) produces an improved slop in the ramp region, corrects for local extrema in the responses of individual measures, and reduces noise, as shown in Fig. 3(b). Another approach is to use different sharpness measures in different regions. When a state transition is issued based on the criteria listed in Table 1, the sharpness measure suitable for the next state is computed. In comparison with the summation method, the computational complexity per iteration is reduced at

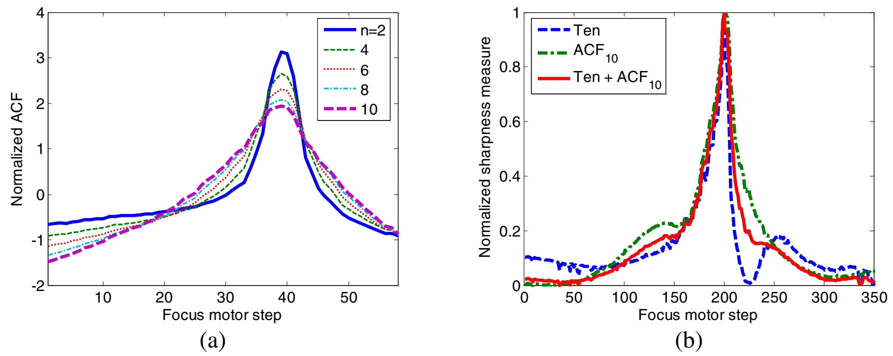


Fig. 3. (a) ACF sharpness measure with various window sizes n for the license plate (LP) sequence (shown in Fig. 4 with a system magnification of 2.28×). (b) Comparison of the Tenengrad (Ten) measure and a linear combination of this measure and the ACF_{10} for the man's face high magnification (MFH) sequence (shown in Fig. 4 with a system magnification of 70×).

the cost of degraded rectification capabilities. The choice of these two methods is application dependent. In our implementation, the summation of two sharpness measures is used for imaging systems with higher magnifications (250×~500×) to make full use of its rectification capabilities while the second approach is employed for imaging systems with lower magnifications (50×~250×) for its lower computational complexity.

5 Experimental Results

To evaluate the performance of various search algorithms, each in conjunction with different sharpness measures, we carried out the following experiments. Images are collected at uniformly distributed camera focus positions and their sharpness measures are computed. A search algorithm is then applied to locate the best focus position. Ideally, the estimated focused position should correspond to the maximum sharpness value. Any difference between them is the estimation error and is expressed in motor steps. The size of the estimation error is translated into the accuracy of the search algorithm. Another performance criterion, the speed of convergence, is described by the number of iterations and the number of motor steps traveled before the optimal focus is obtained.

Four low magnification image sequences (2.28×), resolution chart (RC), Hello-Kitty doll (HD), license plate (LP), and man's face (MFL), are collected by a Canon A80 camera at an interval of 3 focus motor steps covering the 0.2m to infinity focus range with a total of 60 images per sequence. The RC and LP sequences exemplify images with strong and clustered edges. The high magnification image sequences (70×~1500×) are collected by the Sony TRV730 and the Celestron scope. Various system magnifications are used: 70×, 100×, 245×, 500×, and 1500×. At each sampled magnification, two sequences (400 frames per sequence) are collected, one of a scene with strong and clustered edges such as the brick wall (BW) sequence and the other with scattered and low contrast edges such as the man's face (MFH) sequence. Fig. 4 shows sample images from the LP and MFH (70×) sequences, collected at the best focus position and at the end points of the focus range.

Three types of sharpness measures are used: gradient based (Sum Modulus Difference (SMD) [13], Tenengrad (Ten), and Laplacian (Lap)), autocorrelation based (ACF), and frequency domain based (Fast Fourier Transform (FFT) [14] and Frequency Entropy (FE) [15]). Considering accuracy, computational complexity, and invariance to image noise and blur, the RS and FF algorithms present the best performance and are selected as references in comparison with the proposed algorithm. In our implementation, the increments for the *peak*, *ramp*, and *saturation* regions, obtained empirically, are 4, 16, and 32, respectively.

In the interest of space, only the experimental results for the MFH sequence with a magnification of 70× are presented in Fig. 5. Our algorithm achieves accuracy comparable to the RS algorithm. In addition, our algorithm requires a smaller number of both iterations and motor steps. Overall, our algorithm provides a better balance between accuracy and complexity.

Fig. 6(a)-(c) show the sampled frames from a real-time auto-focusing sequence collected at a system magnification of 70×. Fig. 6(d) depicts the sampled

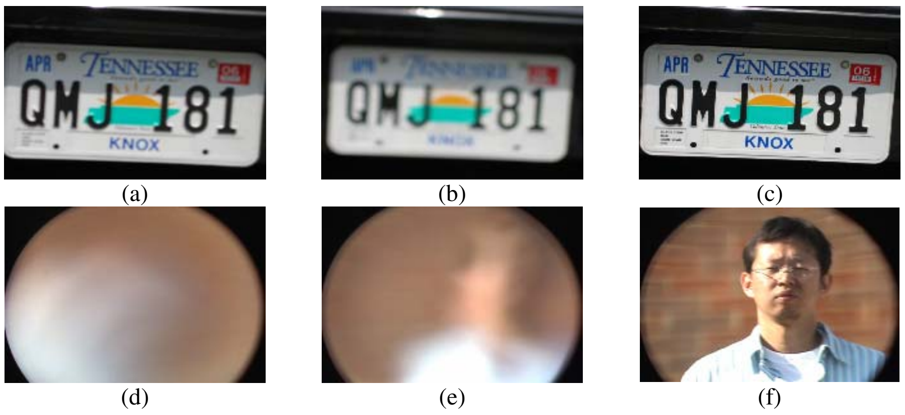


Fig. 4. Sample images from the LP (system magnification: 2.28×, target distance: 1m) and the MFH sequence (system magnification: 70×, target distance: 65m): (a)/(d) far focus end, (b)/(e) near focus end, and (c)/(f) best focus

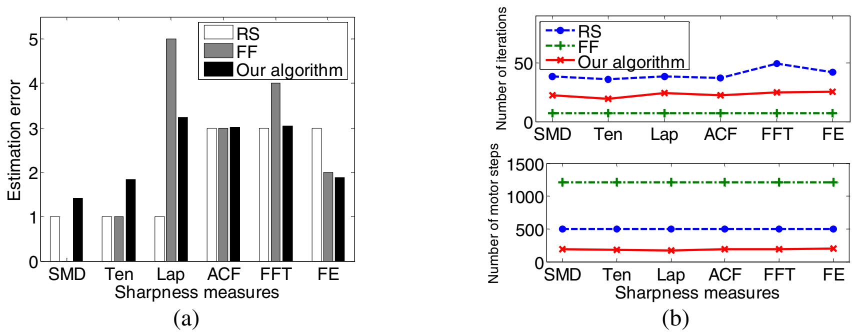


Fig. 5. Comparison across sharpness measures and search algorithms including RS, FF, and our auto-focusing algorithm at 70× magnification. (a) Estimation error. (b) Total number of iterations/motor steps.

focus positions and the corresponding sharpness measures. Given a starting point within ±100 motor steps of the peak region and with a frame rate of 7.2 frames/sec, our algorithm can precisely detect the optimal focus position within 2 seconds.

Exhaustive experiments with various system magnifications and observation distances are conducted to test the effectiveness of our auto-focusing algorithm. Based on raw images, our auto-focusing algorithm works properly for a system magnification of up to 250×. Further increases in magnification result in severely blurred images which undermine the ability of the sharpness measures to produce a smooth and unimodal curve and in consequence lead to possible malfunction of the search algorithm. Image pre-processing and the use of a summation of two types of sharpness measures are possible solutions.

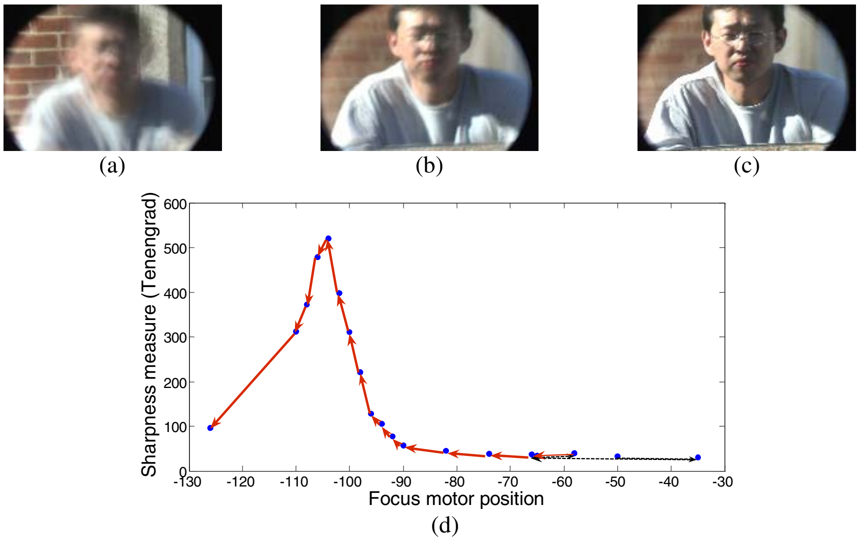


Fig. 6. Sample frames (magnification: 70×, distance: 65m) collected at: (a) Initial focus position, (b) last evaluated focus position, and (c) best focus position. (d) Sampled focus positions. Starting position: -50. Estimated optimal focus position: -102. Motor steps: 106. Time: 1.9s. Dashed lines: direction initialization. Solid lines: search process.

6 Conclusions

For the purpose of long range near-ground surveillance and video tracking, a high magnification imaging system (zoom capability of 70× to 1800×) was built. An image-based passive auto-focusing mechanism, including hardware design and algorithm development, was introduced and applied to long range and high magnification imaging systems. Two strategies, the derivation of the transition criteria and the selection of sharpness measures, were studied to resolve the problems unique to such systems: severe magnification blur and large dynamic focus range. Different from the conventional search algorithms, the transition criteria were derived using ML estimation and well suited to noisy applications. For a faster convergence, autocorrelation and gradient based sharpness measures or their summations were used in different search stages. Experiments based on real-time image sequences verified the effectiveness and overall superiority of our proposed system.

References

1. Collins, R.T., Lipton, A.J., Kanade, T., Fujiyoshi, H., Duggins, D., Tsin, Y., Tolliver, D., Enomoto, N., Hasegawa, O.: A system for video surveillance and monitoring. Technical Report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University (2000)
2. Haritaoglu, I., Harwood, D., Davis, L.S.: W4: Real-time surveillance of people and their activities. IEEE Trans. on Pattern Analysis and Machine Intelligence 22(2000) 809–830

3. Yao, Y., Abidi, B., Abidi, M.: Digital imaging with extreme zoom: system design and image restoration. In: IEEE Conf. on Computer Vision Systems, New York (2006)
4. Krotkov, E.P.: Active computer vision by cooperative focus and stereo. New York: Springer-Verlag (1989)
5. Subbarao, M., Wei, T.: Depth from defocus and rapid autofocusing: a practical approach. In: IEEE Conf. on Computer Vision and Pattern Recognition. (1992) 773-776
6. He, J., Zhou, R., Hong, Z.: Modified fast climbing search auto-focus algorithm with adaptive step size searching technique for digital camera. IEEE Trans. on Consumer Electronics 49 (2003) 257–262
7. Choi, K.S., Lee, J.S., Ko, J.S.: New autofocusing technique using the frequency selective weighted median filter for video camera. IEEE Trans. on Consumer Electronics 45(1999) 820–827
8. Subbarao, M., Tyan, J.K.: Selecting the optimal focus measure for autofocusing and depth-from-focus. IEEE Trans. on Pattern Analysis and Machine Intelligence 20(1998) 864–870
9. Kehtarnavaz, N., Oh, H.J.: Development and real-time implementation of a rule based auto-focus algorithm. Journal of Real-Time Image 9 (2003) 197–203
10. Yao, Y., Abidi, B., Abidi, M.: Evaluation of sharpness measures and search algorithms for the auto-focusing of high magnification images. In: SPIE, Orlando, FL (2006)
11. Lee, J.H., Kim, K.S., Nam, B.D., Lee, J.C., Kwon, Y.M., Kim, H.G.: Implementation of a passive automatic focusing algorithm for digital still camera. IEEE Trans. on Consumer Electronics 41 (1995) 499–454
12. Batten, C. F.: Autofocusing and astigmatism correction in the scanning electron microscope. Master's thesis, University of Cambridge (2000)
13. Santos, A., Ortiz de Solorzano, C., Vaquero, J. J., Pena, J. M., Malpica, N., del Pozo, F.: Evaluation of autofocus functions in molecular cytogenetic analysis. Journal of Microscopy 188(1997) 264-272
14. Chern, N. K., Neow, P. A., Ang, M. H.: Practical issues in pixel-based autofocusing for machine vision. In: Int. Conf. on Robotics and Automation, Seoul, Korea (2001) 2791-2796
15. Kristan, M., Pers, J., Perse, M., Kovacic, S.: A Bayes-spectral-entropy-based measure of camera focus using a discrete cosine transform. Pattern Recognition Letters 27(2006) 1431-1439

Trifocal Transfer Based Novel View Synthesis for Micromanipulation

Julien Bert, Sounkalo Dembélé, and Nadine Lefort-Piat

Laboratoire d'Automatique de Besançon
UMR CNRS 6596 - ENSMM - UFC
25000 Besançon, France
{jbert, sdembele, npiat}@ens2m.fr

Abstract. In trifocal transfer based novel view synthesis, matched pixels of both input views are projected in the novel view. The angle of view of this latest is usually narrow, i.e. the novel view is very close to input ones. In this paper we improve the method to get a large angle of view. A simplex approach is used to compute the model of the virtual views pose. This model allows the computation of the novel view at any desired angle of view. We also show that those results are very useful in micromanipulation tasks where transfer of edges is enough instead of the entire pixels of input views.

1 Introduction

Novel View Synthesis (NVS) is a part of computer vision introduced by [1]. It deals with the obtaining of a maximum of views of an environment from a minimum of real data on it. For example, a lateral view of an object can be synthesized with only two real top views. There are two classes of methods in NVS: the model-based rendering and the image-based rendering.

In model-based rendering (MBR), virtual environments are created from mathematical models. A typical example is 3D characters synthesis in movies and video games by modeler softwares. In image-based rendering (IBR), a set of real images of the scene is used to build a novel view. According to the knowledge about scene geometry, [2] proposes the following classification: rendering with no geometry, rendering with explicit geometry and rendering with implicit geometry. Rendering with no geometry i.e. no calibration is used to create a mosaic from a set of local views that leads to a novel global view [3], [4]. Rendering with explicit geometry i.e. with strong calibration is close to MBR. Its purpose is the reconstruction of a 3D view from real views of the scene [5]. This technique needs a strong calibration and is computationally expensive. Rendering with implicit geometry only needs a weak calibration. Ref. [6], presents three techniques of NVS of this type: the line of sight, the epipolar transfer and the trifocal transfer. The line of sight approach is based on ray-tracing [7]. Its drawback is the fact that at least ten images are required to obtain a synthetic view. The epipolar transfer approach is introduced by [8], it is based on epipolar geometry where the epipolar constraint defines the point-line duality in pair of images: one point in the left view corresponds to a line in the right view. This concept is used to create a virtual view from two real views. Each point of the virtual view is the intersection of the lines of the points from the real views. The trifocal transfer approach first proposed by [9] is based on the trifocal constraint between three views. The later is defined by a tensor. With two real images and a tensor, all the points of the images are transferred into a novel view. The control of that corresponding virtual view pose is performed through a transformation matrix.

This virtual view can be very useful in micromanipulation which concerns the manipulation of parts at the microscale, i.e. in the range from 1 μm to 1 mm. The main applications of micromanipulation are assembly, sorting and testing of microparts. In addition to biomicroparts like cells and pollen seeds, artificial microparts are chemically or mechanically synthetized, or micromachined. Classical examples of the first and second types are respectively grains of powder like drugs or cosmetics, and optomechatronic components like balls, pegs, pins, threads, membranes, lenses, shutters and fibres. In some cases these microparts define final products (MEMS), otherwise they must be assembly to lead to the final products. For that purpose some automated microassembly systems have been developed by [10], [11], [12] and [13]. From those results it can be noticed that a microimaging system is always required, and the most used is a photon microscope connected to a camera. The images and their processing and analysis allow the task surveillance, system control or microparts recognition. The field-of-view of the microscope is very narrow that leads to the use of multiple views imaging : global view (usually at the top), left and right lateral views. The second reason of multiple views use is the fact that top view only allows the access to the xy position of the microgripper. Lateral view is required to get the z position. The third reason is the occurrence of components occlusions during assembly, the microgripper can hide the microparts to pick. However multiple views imaging has a drawback, the microimaging component cannot be positioned anywhere, so some views are not accessible. Sometimes, it is also useful to set free the work field. A view from a virtual imaging system using a novel view synthesis method can overcome that problem.

In this paper we use a trifocal approach without explicit 3D data to synthesize a virtual view that can be very useful in micromanipulation. In Image Based Rendering (IBR) literature, the novel view is close to real views, but in this paper we extrapolate the angle of view up to 85^o. Section 2 summarizes the trifocal geometry and describes it use to generate a novel view. Section 3 presents a new method to automatically obtain the angle of view wanted. Section 4 presents experimental results.

2 Trifocal Transfer

The trifocal transfer is the method of IBR with implicit geometry. It only requires a weak calibration which implies the estimation of the fundamental matrix. Trifocal transfer is based on the geometry of three views, named the trifocal geometry.

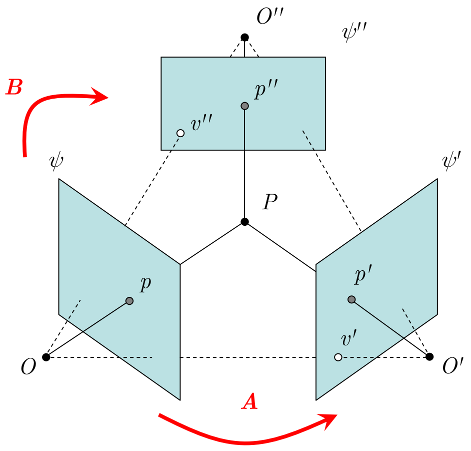


Fig. 1. Trifocal geometry

2.1 Trifocal Geometry and Trilinear Tensor

Trifocal geometry is the extension of epipolar geometry to three views. Let us consider three views of $\mathcal{P}^2$ ψ, ψ' and ψ'' (Fig. 1). A point $P \in \mathcal{P}^3$ is projected onto the point $p = (x, y, 1)^T$ in ψ, $p' = (x', y', 1)^T$ in ψ' and $p'' = (x'', y'', 1)^T$ in ψ''. Let us note:

- A and B the collineation matrixes corresponding respectively to the projective transformations $\psi \rightarrow \psi'$ and $\psi \rightarrow \psi''$,
- v' and v'' the epipoles i.e. the projection of the optic center O on respectively ψ' and ψ''.

The trilinearity defines the constraint between three views [14]: p, p' and p'' are linked by the same projected point P. The epipolar geometry of (ψ, ψ') and (ψ, ψ'') allow to write the following equations:

$$\begin{aligned} p' &\cong Ap + \delta v' \\ p'' &\cong Bp + \delta v'' \end{aligned} \tag{1}$$

where δ is the relative affine structure of P. The coefficient δ is independent of ψ', i.e., is invariant according to the choice of the second view [15]. Then δ can be isolated from both (1) to obtain a set of equalities. From those equalities trilinear equations linking p, p' and p'' can be recovered: four linearly independent equations with 27 distinct coefficients are obtained. Each of these is an element of the trilinear tensor $\mathcal{T}_i^{jk}$ $i, j, k \in [1, 3]$:

$$\mathcal{T}_i^{jk} = v'^j b_i^k - v''^k a_i^j \tag{2}$$

where a_i^j and b_i^k are the elements of the collineation matrix A and B with $i, j, k \in [1, 3]$ (i is the index of the column, j is the index of the rows and k is

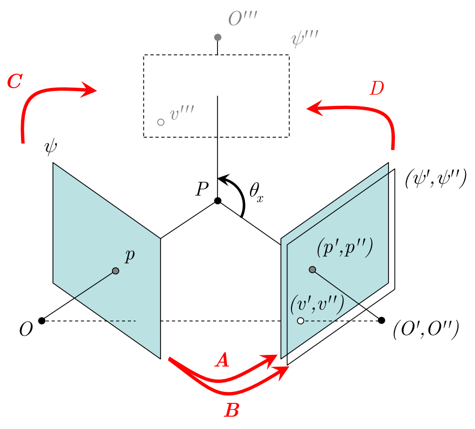


Fig. 2. The principe of novel view synthesis from two input views

the index of the layer). The trilinear tensor $\mathcal{T}_i^{jk}$ is a $3 \times 3 \times 3$ array of the 27 trilinear coefficients. Then the four trilinear equations, can be written with the tensor:

$$\begin{array}{l} x''\mathcal{T}_i^{13}p^i - x''x'\mathcal{T}_i^{33}p^i + x'\mathcal{T}_i^{31}p^i - \mathcal{T}_i^{11}p^i = 0 \\ y''\mathcal{T}_i^{13}p^i - y''x'\mathcal{T}_i^{33}p^i + x'\mathcal{T}_i^{32}p^i - \mathcal{T}_i^{12}p^i = 0 \\ x''\mathcal{T}_i^{23}p^i - x''y'\mathcal{T}_i^{33}p^i + y'\mathcal{T}_i^{31}p^i - \mathcal{T}_i^{21}p^i = 0 \\ y''\mathcal{T}_i^{23}p^i - y''y'\mathcal{T}_i^{33}p^i + y'\mathcal{T}_i^{32}p^i - \mathcal{T}_i^{22}p^i = 0 \end{array} \tag{3}$$

2.2 Novel View Synthesis by Trilinear Tensor

The first application of trilinear tensor to NVS is reported in [9] where three real views are used to compute the trifocal tensor and the virtual view: three real views lead to a virtual view. Later the authors proposed in [16] a more subtle approach that consists in merging two of the three input views: as a result, a novel view (virtual) is obtained from two real views.

Let us consider three views ψ, ψ' and ψ''. As explained above the trilinear tensor $\mathcal{T}(\psi, \psi', \psi'')$ can be calculated by (2). Now suppose ψ'' is merged with ψ' (Fig. 2). That means the collineation matrixes A $(\psi \rightarrow \psi')$ and B $(\psi \rightarrow \psi'')$ and the epipoles v' and v'' are identical. Thus (2) becomes:

$$\mathcal{T}_i^{jk} = v'^j a_i^k - v'^k a_i^j \tag{4}$$

This latest defines what is called the seed tensor. Let us suppose the view ψ'' becomes the view ψ''' by a collineation matrix D, then a collineation matrix C links ψ and ψ'''. As the same, the seed tensor $\mathcal{T}(\psi, \psi', \psi'')$ changes to $\mathcal{G}(\psi, \psi', \psi''')$. As $C = DB$ the new tensor $\mathcal{G}$ can be computed from (2) and (4):

$$\mathcal{G}_i^{jk} = d_l^k \mathcal{T}_i^{jl} + t^k a_i^j \tag{5}$$

where $t^k = d_l^k v''^k - v'''^k$ is the element of the translation vector t that changes $v'' \rightarrow v'''$ and d_l^k is the element of collineation matrix of D, with $i, j, k, l \in [1, 3]$. Every point p''' of ψ''' can be calculated by (3):

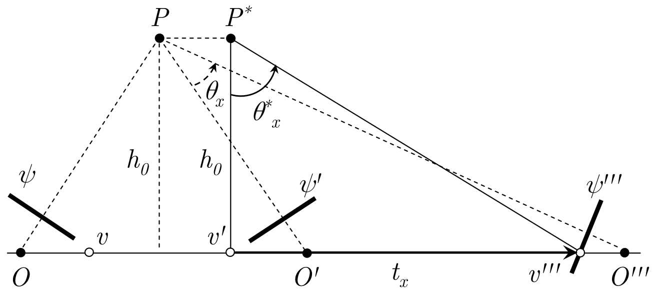


Fig. 3. The angle of view of ψ''' according to the translation t_x

$$x''' = \frac{x'\mathcal{G}_i^{31}p^i - \mathcal{G}_i^{11}p^i}{x'\mathcal{G}_i^{33}p^i - \mathcal{G}_i^{13}p^i} \qquad y''' = \frac{x'\mathcal{G}_i^{32}p^i - \mathcal{G}_i^{12}p^i}{x'\mathcal{G}_i^{33}p^i - \mathcal{G}_i^{13}p^i} \tag{6}$$

The process requires the weak calibration of the imagers and the points correspondence between both input views.

2.3 Computation of the Angle of View

We have exposed above how to synthesize a virtual view using trilinear tensor approach. In practice, the view ψ''' does not exist. It synthesis from ψ and ψ' requires the computation of the tensor element $\mathcal{G}_i^{jk}$ which is a function of two sets of parameters d_l^k and t^k. In order to simplify the synthesis D will be set equal to the identity matrix $I_{3\times3}$. The translation vector is a function of the angle of view, $t = f(\theta)$, defined by the angle between the lines $[PO']$ and $[PO''']$ in the trifocal plane. We only use the component t_x and suppose it depends on θ_x. Thus the problem is to find the value of t_x for a given value of θ_x.

The value of θ_x is approximated by θ_x^*, the angle between the lines $[P^*v']$ and $[P^*v''']$ where $[PP^*]$ is parallel to $[OO']$ and $[P^*v']$is perpendicular to $[OO']$ (Fig. 3). Then t_x can be written:

$$t_x = h_0 \tan \theta_x^* \tag{7}$$

In order to estimate h_0 we create a dummy segment in a plane of $\mathcal{P}^3$ parallel to ψ' which we project in the view ψ'''. At $\theta_x^* = 0$ and then $t_x = 0$, the length of the pattern in ψ''' is L_0, and at $\theta_x^* \neq 0$ then $t_x \neq 0$ the length become $L(t_x)$. We can write:

$$L_0 \cos \theta_x^* = L(t_x) \tag{8}$$

That equation can be solved using a Nelder-Mead simplex method [17]. That optimization method compares the values of the objective function with zero and does not require the use of any derivatives. A simplex in $\mathbb{R}^n$ is a set of $n+1$ points that do not lie in a hyperplane. For example a triangle is a simplex of 2 dimensions. In the Nelder-Mead method, the simplex can vary in shape from iteration to iteration following reflect, expand, contract and shrink. The simplex finds the minimal of (8) according to t_x. As soon as t_x and θ_x^* are known, h_0

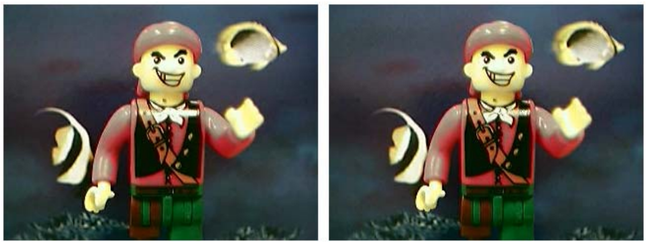


Fig. 4. Our stereo images

can be computed according to (7). Thus the model of displacement is entirely defined and can be used for the computation of the tensor and then the novel view.

3 Experimental Results

The principles exposed above are used to compute a novel view, a lateral one, from two front views (Fig. 4).

3.1 Making the Dummy Pattern

This process requires the calculation of the disparity interval i.e. the displacement between every (left and right) couple of points. A Canny [18] is used to compute the edges and the matching is achieved by Zhang method [19] with a Sum of Squared Differences (SSD) correlation criterion. Usually the correspondent of the point p of the left image in the right one (p') is searched along the epipolar line. The edges transfer is enough for our experiment since our application is the surveillance and control of micromanipulation task. Complete images

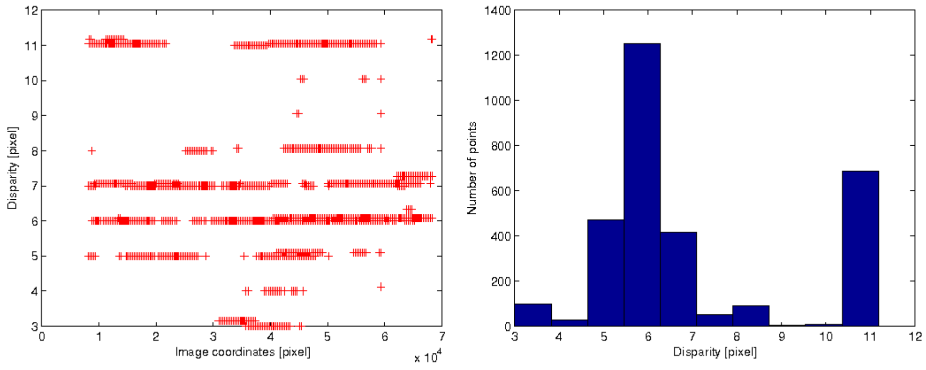


Fig. 5. Left, the layer representation of the disparity edge map between images. Right, distribution of disparity.

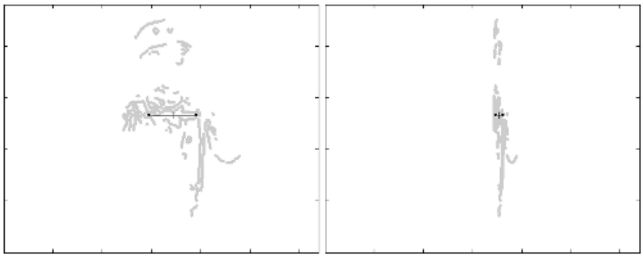


Fig. 6. Left, the master layer with the dummy pattern for $t_x = 0$. Right, the same with $t_x = 900000$.

with texture and color are not necessary. At the end the correspondence of every point of the edges is achieved and the edge-map disparity is computed (Fig. 5). As exposed above the dummy pattern is inlayed in the virtual view and allows the computation of the displacement vector of the view. All the points of the pattern must be on the same plane i.e. at the same layer. For the later we choose the one for which the number of points is maximal.

Let us note Ω the set of points. Since the novel view displacement is reduced to θ_x (see above) Ω can be divided perpendicularly to x axis into left Ω_L and right Ω_R sets of points. The centroid of the three sets are computed and they coordinates are used to define the following points: (x_L, y), (x, y) and (x_R, y) where x, x_L and x_R correspond respectively to the x coordinate of the centroid of Ω, Ω_L and Ω_R and y corresponds to the y coordinate of Ω. Those points define the dummy pattern (Fig. 6).

3.2 Computing the View

It is impossible to compute the view for $\theta_x = 90^o$, in is this case t_x trends toward infinity according to (7). So, for the lateral view we choose a maximum angle of view of 85^o. For an arbitrary angle of view of $\theta_x^* = 70^o$, the simplex method leads to a t_x of 255943.

Figure 7 shows the length of the pattern versus t_x and the angle of view (model and result) versus t_x. According to (7), the value of h_0 is 68580. Finally, for the view at 85^o, the displacement vector is $t_x = 68580 \tan(\theta_x)$. Where the value of t_x is known, the lateral view is computed in real time with the two input views using (6).

But, that is not sufficient to ensure the quality of the view. The points are also rectified by minimizing their shift and maintaining the center of the pattern at the center of the view. Figure 8 shows eight lateral views from 0^o to 85^o angle of view.

3.3 Application to Micromanipulation

We apply above principles to a microassembly scene: the picking up of a microgear by a microgripper. The lateral view, at 85^o angle of view, allows the

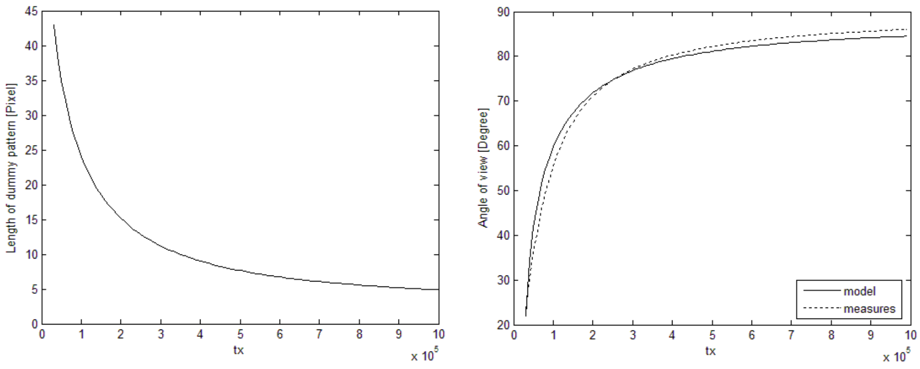


Fig. 7. Left, the size of the dummy pattern according to t_x. Right, comparison between model and experimental measures of the angle of view according to t_x.

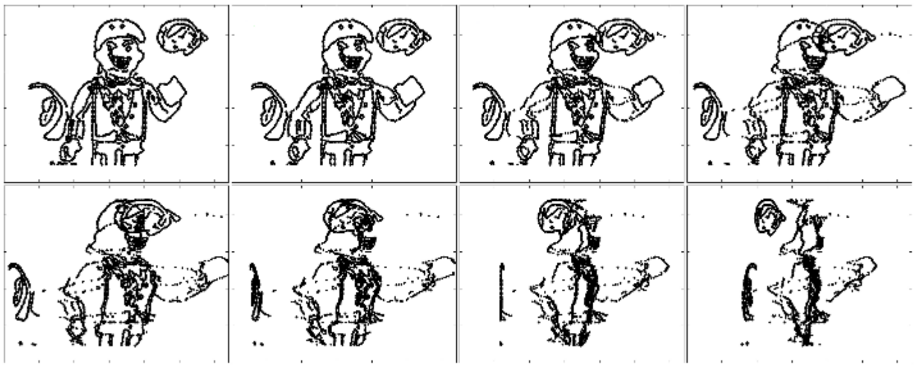


Fig. 8. Lateral views where the angle of view increases from 0^o to 85^o

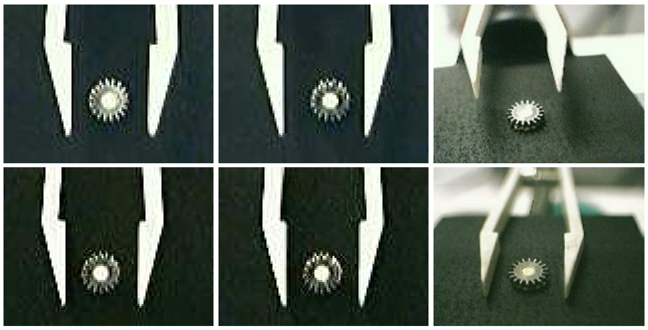


Fig. 9. Top, stereo top images and lateral image of the scene with the microgear outside the gripper tips. Bottom, the same scene with the microgear inside the gripper tips.

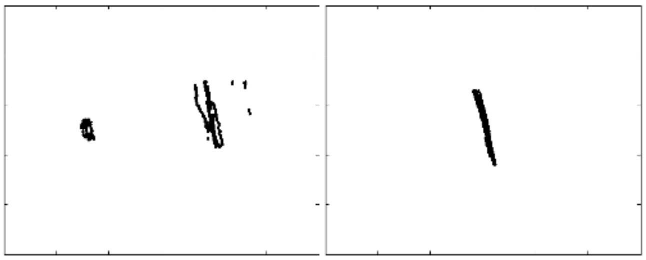


Fig. 10. The views at 85^o according to views of Fig. 9. Left corresponds to top views, right to bottom views.

access to the z position of the gripper according to the microgear. Two cases are considered: the part is not between the tips and the part is between the tips (Fig. 9).

Figure 10 shows the views at an angle view of 85^o for the two configurations. In spite the weak number of layers, it is possible to evaluate the distance between the gripper and the microgear. Then it is possible to know if the gear can be picked up or not.

4 Conclusion

We summarized trifocal geometry and explained how it allows the expression of trifocal constraints through trilinear tensor. This latest is required to transfer matched pixels in both input views into the virtual one. We quickly computed from the measure of the length of a dummy segment in a novel view and the Nelder-Mead simplex method, the model of the novel views pose. That model allows the computation of novel views with very large angle of view.

We applied that approach to images from a micromanipulation scene and showed that the obtained image of edges is enough to ensure the surveillance of the task.

Future work will deal with the deepening of the modelisation of the views pose and it application to synthesize virtual imagers in micromanipulation. The great merit of that idea is it will set free the work field.

References

1. Chen, S.E., Williams, L.: View interpolation for image synthesis. In: Computer Graphics (SIGGRAPH'93 Proceedings). (1993) 279–288
2. Shum, H.Y., Kang, S.B.: A review of image-based rendering techniques. In: SPIE Proceedings of the Visual Communications and Image Processing. (2000) 2–13
3. Chen, S.: Quicktime vr - an image-based approach to virtual environment navigation. In: Computer Graphics (SIGGRAPH'95 Proceedings). (1995) 29–38
4. Szeliski, R., Shum, H.Y.: Creating full view panoramic image mosaics and environment maps. In: Computer Graphics (SIGGRAPH'97 Proceedings). Volume 31. (1997) 251–258

5. Saito, H., Baba, S., Kanade, T.: Appearance-based virtual view generation from multicamera videos in the the 3-d room. In: IEEE Transactions on Multimedia. Volume 5. (2003) 303–316
6. Connor, K., Reid, I.: Novel view specification and synthesis. In: Proceeding of the British Machine Vision Conference, Cardiff, England (2002) 243–252
7. Irani, M., Hassner, T., Anandan, P.: What does the scene look like from a scene point? In: Proceedings of European Conference on Computer Vision (ECCV), Copenhagen (2002) 883–897
8. Faugeras, O., Robert, L.: What can two images tell us about a third one? Technical report 2018, INRIA (1993)
9. Avidan, S., Shashua, A.: Novel view synthesis in tensor space. In: IEEE Computer Vision and Pattern Recognition. (1997) 1034–1040
10. Yang, G., Gaines, J.A., Nelson, B.J.: A surpervisory wafer-level 3d microassembly system for hybrid mems fabrication. Journal of Intelligent and Robotic Systems **37** (2003) 43–68
11. Matsumoto, A., Akimoto, T., Yoshida, K., Inoue, H., Kamijo, K.: Development of mems component assembly machine - application of robotics technology to micromechatronics. In: The International Symposium on Micro-Mechanical Engineering. (2003) 83–88
12. Popa, D.O., Stephanou, H.E.: Micro and mesoscale robotic assembly. Journal of Manufacturing Process **6** (2004) 52–71
13. Sun, L., Xie, H., Rong, W., Chen, L.: Task-reconfigurable system for mems assembly. In: IEEE International Conference on Robotics and Automation, Barcelona, Spain (2005) 844–849
14. Shashua, A.: Algebraic functions for recognition. In: IEEE Transactions on Pattern Analysis and Machine Intelligence. Volume 17. (1994) 779–789
15. Shashua, A., Navad, N.: Relative affine structure: Theory and application to 3d reconstruction from perspective views. In: IEEE Computer Vision and Pattern Recognition (CVPR). (1994) 483–489
16. Avidan, S., Shashua, A.: Novel view synthesis by cascading trilinear tensors. In: IEEE Transactions on Visualization and Computer Graphics (TVCG). Volume 4. (1998) 293–306
17. Nelder, J.A., Mead, R.: A simplex method for function minimization. Computer Journal **7** (1965) 308–313
18. Canny, J.F.: A computational approach to edge detection. In: IEEE Transactions on Pattern Analysis and Machine Intelligence. Volume 8. (1986) 679–698
19. Zhang, Z., Deriche, R., Faugeras, O., Luong, Q.T.: A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. Technical report 2273, INRIA (1995)

Simulation of Diabetic Retinopathy Neovascularization in Color Digital Fundus Images

Xinyu Xu[1], Baoxin Li[1], Jose F. Florez[2,3,4], and Helen K. Li[2,3,4]

[1] Dept. of Computer Science and Engineering, Arizona State University, Tempe, AZ, U.S.A
[2] Dept. of Ophthalmology and Visual Sciences, The University of Texas Medical Branch, Galveston, TX, U.S.A
[3] School of Health Information Sciences, University of Texas Health Science Center, Houston, TX, U.S.A
[4] Universidad De Antioquia, Medellin, Colombia, U.S.A

Abstract. Diabetic retinopathy (DR) has been identified as a leading cause of blindness. One type of lesion, neovascularization (NV), indicates that the disease has entered a vision-threatening phase. Early detection of NV is thus clinically significant. Efforts have been devoted to use computer-aided analyses of digital retina images to detect DR. However, developing reliable NV detection algorithms requires large numbers of digital retinal images to test and refine approaches. Computer simulation of NV offers the potential of developing lesion detection algorithms without the need for large image databases of real pathology. In this paper, we propose a systematic approach to simulating NV. Specifically, we propose two algorithms based on fractal models to simulate the main structure of NV and an adaptive color generation method to assign photorealistic pixel values to the structure. Moreover, we develop an interactive system that provides instant visual feedback to support NV simulation guided by an ophthalmologist. This enables us to combine the low level algorithms with high-level human feedback to simulate realistic lesions. Experiments suggest that our method is able to produce simulated NVs that are indistinguishable from real lesions.

1 Introduction

Diabetic retinopathy (DR) has been identified as a leading cause of blindness [1]. Studies have shown that early detection and treatment significantly reduces the risk of severe vision loss [2]. Diabetic retinopathy evaluation programs typically rely on experts to review large numbers of retinal images from diabetic patients. Researches are underway to use computers to assist or automatically detect/diagnose DR by analyzing color digital fundus images [3-7]. While computer-assisted approaches offer the possibility of more cost-effective or timelier evaluation, they also present new challenges. For example, how algorithms are affected by differences in digital imaging factors such as resolution, contrast or color is not well understood. Testing detection/diagnostic algorithms also requires what may be prohibitively large image databases of real DR lesions. Computer-generated lesions offer the possibility of testing DR detection/diagnostic algorithms on large simulated datasets and tuning approaches to account for differences in digital imaging factors.

In this paper, we develop techniques to simulate neovascularization (NV), a common DR lesion that signifies the disease has reached a vision-threatening phase. NV is a growth of new blood vessels on the surface of the retina [8] (Fig. 2 left). A review of literature indicates that little work has been done in simulating human tissue in digital imagery. The few examples include the work done by Landini for simulating corneal neovascularization [9] and by Hoe for liver lesion simulation [10].

In our work, we propose a systematic approach to simulating NV. Two algorithms based on local fractal growth models are proposed to simulate the geometrical structure of an NV lesion. A color generation method, which is adaptive to the region to which the simulated NV is inserted, is then proposed for assigning photorealistic pixel values to the structure. Moreover, an interactive system is developed for providing instant visual feedback to support NV simulation guided by an ophthalmologist. This enables us to combine the low level simulation algorithms with high-level human feedback to generate realistic NVs. Our current experiments on non-proliferative DR images have generated NVs that are deemed realistic by ophthalmologists. The complete system is under deployment for ophthalmologists' formal evaluation of its performance including the acceptance rate (see Sect. 3).

2 Proposed Method

2.1 Methodology Overview

Fractal geometry is commonly encountered in nature, e.g., branching patterns in trees, blood vessels patterns and shape of tumors studied in pathology. Fractals are based on the concept of self-similarity of spatial geometrical patterns despite a change in scale or magnification so that small parts of the pattern exhibit the pattern's overall structure [17]. The concept of fractals as mathematical entities to describe complex natural branching patterns was first considered by Mandelbrot [12]. The fractal dimension (D), typically a non-integer value between 1 and 2, describes how thoroughly the pattern fills two-dimensional spaces [11].

The applications of fractals to biology and medicine cover a wide range of scale: molecules, cells, tissues, and organs [18]. Masters and Platt [19] and Family et al. [20] were the first to introduce the use of fractal analysis to retinal vascular branching patterns. One common goal of these studies is to determine the fractal dimension of those structures and then to use this number as an index to discriminate the class of normal structures from abnormal and pathological structures [13].

Inspired by the work on fractal-based *analysis*, we propose to do fractal-based *synthesis*, NV simulation. Our objective is to *create* by simulation NVs that conform to bio-physical growth mechanism of real NVs and are consistent with the observed appearance of real NVs. This is a challenging task. Some of the key challenges that affect the morphology and appearance of NV and our corresponding strategies are discussed in the following:

1. NV could present various *patterns*, which are mostly like random winding vessels and some may be like flowers, sea coral or other complicated structures. While there is no proven optimal way of simulating these patterns, inspired by the success of fractal-based analysis of retina vessels, we employ three fractal models to create

the structure of NV: random walk [21] (self-avoiding and self-intersecting), invasion percolation (IP) [14] and spreading percolation (SP) [14]. The random walk models are chosen with the consideration that the NVs are random in nature and thus no models with strong structural constraint should be used. We will present in detail the random walk fractal model.

2. The *caliber* of NVs is much smaller than the natural retinal veins from which NVs originate. In our simulation, the caliber of NV is a function of the width of the retinal vein branch.
3. The *colors* of NV in most cases are reddish varying with different degree of saturation and brightness. We generate colors by sampling the empirical hue/saturation/value (HSV) color density defined by local normal vessel segments.
4. NVs are usually *located* at the connection of branches of natural retinal veins or at arterial-venous crossing sites. In our simulation, an ophthalmologist specifies optimal locations where simulated NV should be inserted using a graphic user interface. This interface also allows other parameters, such as coverage and complexity of the simulated NV, to be configured.

The key steps of the approaches are illustrated in Fig. 1, with the details of the algorithms presented in subsequent sections.

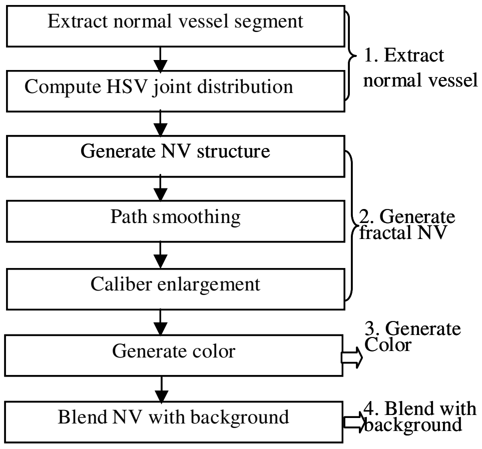


Fig. 1. DR NV simulation architecture

2.2 NV Shape Generation

Generate Binary NV Structure Using Fractals. Our first algorithm for NV structure simulation is based on the following observation: NVs are thin, long, connected, winding vessels with variable degrees of random curvature. To simulate these vessels, we designed an algorithm whose core is a self-intersecting random walk.

ALGORITHM 1:

1. Create a square lattice with side length *2L+1* and spacing 1, initialize the center of the lattice, *O*, to be occupied by a particle.

2. The first position the particle jumps to (x_1) is randomly chosen on a circle with radius r and centered at O. Suppose the angle at position x_1 is θ_1, then $\theta_1 = 2*pi*rand$.
   Loop for $t = 2{:}TIMES$
3. Along the direction pointed out by θ_{t-1}, we create a circular sector centered at x_{t-1} with the central angle 2ε and radius r (ε denotes the value of half central angle). The position of the walking particle at time t, x_t, is randomly chosen on this circular sector given by

$$\begin{aligned} \theta_t &= 2\varepsilon \bullet rand + (\theta_{t-1} - \varepsilon) \\ \mathrm{x}_t^h &= \mathrm{x}_{t-1}^h + r \bullet \cos(\theta_t) \\ \mathrm{x}_t^v &= \mathrm{x}_{t-1}^v + r \bullet \sin(\theta_t) \end{aligned} \quad . \tag{1}$$

   where x_t^h and x_t^v denote the respective horizontal and vertical coordinate of the walking particle at time t.
4. Set lattice position x_t to 1indicating that this site has been occupied.
5. Record the path and the order.

End loop

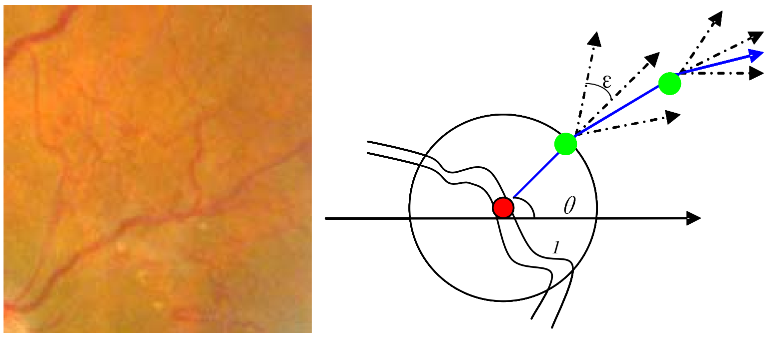


Fig. 2. Left: A DR image with natural NV (thin vague vessels). Right: NV structure generation using *ALGORITHM 1*. The red (dark grey) circle indicates the seed particle at the center of lattice. The green (light grey) circles denote the position of the walking particle at different times. The solid line linking the circles illustrates the growth path of the particle. The dotted lines illustrate the restriction of the growth direction within [$\theta_{t-1}+\varepsilon$ $\theta_{t-1}+\varepsilon$].

Fig. 2(right) graphically illustrates *ALGORITHM 1*. The entire process is a Markovian self-intersecting random walk [21] since the position at time t only depends on the position at time t-1. The path may intersect with itself, which entails some complex patterns. The curvature of the path is controlled by the central angle 2ε of the circular sector. The greater the central angle, the more likely the path is convoluted. Because the pixels of real NV are connected and continuous, the jump distance at each time instance, r, is set to 1 to prevent holes or discontinuity in the generated vessel. In the algorithm, the parameter *TIMES* and the lattice side length L control how much area the NV will cover. Two NV paths generated by this algorithm are shown in Fig. 3 (a).

Our second algorithm is based on the observation that some NVs appear to have the following pattern: (1) from a single seed point located on retinal veins grows one or multiple vessels (we call the major vessel the first level); (2) from major vessels grow one or multiple ramifications that could intersect with each other (we call these ramifications the second level; there could be additional levels); (3) each vessel branch could be a simple curve or very convoluted.

To generate NVs with the above characteristics, we proposed the following self-avoiding random walk algorithms.

ALGORITHM 2:

```
1. Initialize the center of a lattice (side length $2L+1$) to be occupied by a seed particle.
2. The 2-D area surrounding the seed particle is divided into 8 sectors, each representing a possible area the walking particle will grow into (from $(\pi/4)*(j-1)$ to $(\pi/4)*j, j = 1, 2, 3,\ldots 8$).
3. An 8 dimensional probability vector $p=\{p_1, p_2, p_3, \ldots, p_8\}$ is generated where $p_i$ denotes the probability of growing into sector $i$. $p$ is calculated to have one dominant entry $p_j$ which is larger than other probability members so that the particle will more likely grow into area $j$.
   Loop for $t = 1 : TIMES$
4. By sampling the cumulative probability of vector $p$, an angle $\theta_t$ is calculated to determine the position of the walking particle at time $t$. $\theta_t$ is uniformly distributed within $[\pi*(j-1)/4 \quad \pi*j/4]$.
```

$$\begin{aligned} &Direction\,(t) = \textit{the index of pie by sampling } p \\ &\theta_t = (\pi/4)\bullet rand + (\pi/4)\bullet Direction(t); \end{aligned} \quad . \tag{2}$$

```
5. The position of the walking particle at time $t$, $\mathbf{x_t}$, is given by:
```

$$\begin{aligned} x_t^h &= x_{t-1}^h + r\bullet\cos(\theta_t) \\ x_t^v &= x_{t-1}^v + r\bullet\sin(\theta_t) \end{aligned} \quad . \tag{3}$$

```
End loop
```

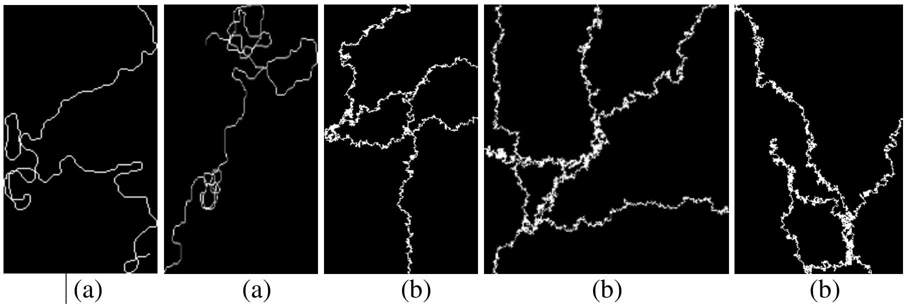

(a) (a) (b) (b) (b)

Fig. 3. (a) NVs generated by self-intersecting random walk (*ALGORITHM 1*). (b) NVs generated by multi-level self-avoiding random walk (*ALGORITHM 2*).

Since one entry of the probability vector p, p_j, is larger than other entries, the vessel usually grows toward one dominant direction with certain local randomness and intersections along the path. Therefore globally this single vessel branch is a self-avoiding random walk. Note that the above steps can only generate one branch. To produce NV with multiple levels and branches that intersect with one another, the algorithm is recursively called with a different *p* for different branches such that the orientations of branches are randomly distributed. Examples of NV structures generated by recursively executing *ALGORITHM 2* are illustrated in Fig. 3 (b).

Path Smoothing. The vessels in Fig. 3 (b) may be too jagged, and thus a smoothing filter is applied to make the path more natural. This is achieved by a moving average along individual paths. Fig. 4 shows results corresponding to Fig. 3 (b) after path smoothing.

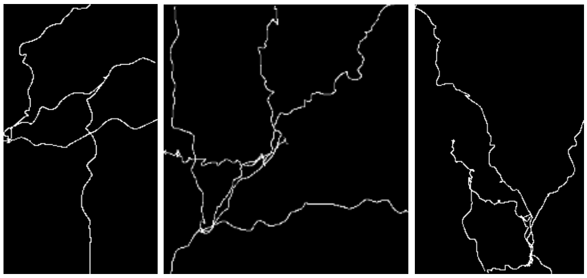


Fig. 4. Random walk fractals after path smoothing

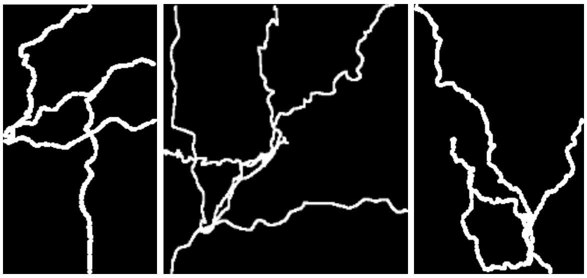


Fig. 5. Binary NV structure after caliber enlargement

Caliber Enlargement. The path generated from the above algorithms may be too thin as the width of path is only one-pixel. To obtain vessels of different calibers, we perform caliber enlargement. The first step is to decide the enlargement scale for each pixel. In the real NV, we found that the wider the normal retina vessel where NV sprouts out, the wider the newly grown NV vessel. Therefore, the enlargement scale is determined by measuring the width of normal vessels where simulated NV will be inserted. Based on the measured width of normal vessel, empirical rules are set to define the enlargement scale of simulated NV vessels: if the width of normal vessel is larger than 10 pixels, the enlargement scale is set to 2 or 3; if the width of normal vessel is larger than 5 but less than 10, the enlargement scale is set to 1 or 2; in other cases, the enlargement scale is 1. Only three scales are used as real NV vessels are usually very thin. Next, for each pixel on a branch, a "disk" is created with the center at the current pixel and a radius equal to the enlargement scale w_i. The values of all pixels within this disk are set to 1, denoting that the pixels have been added to the original NV structure. The disk is then added to the original binary NV using logic operation OR. Using a disk gives us smooth vessels. Fig. 5 illustrates the results of caliber enlargement.

2.3 Photorealistic Color Generation

The binary NV structure such as those in Fig. 5 needs to be ‘painted’ with appropriate colors based on the context of the background. As colors of real NV are similar to the colors of nearby normal vessels, the colors of the simulated NV are generated adaptively by sampling color densities of nearby normal vessel segments.

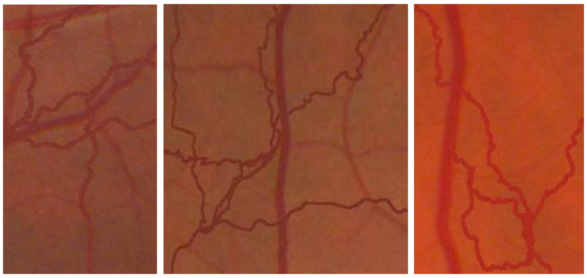


Fig. 6. NV with color overlaid on the background image

Extract Normal Vessel. The extraction algorithm starts from a pixel on a normal vessel where the simulated NV is to be inserted. This pixel may be randomly selected after automatic vessel detection, or more preferably, specified by a user (see Section 3). Then through breadth-first search we find all the pixels with colors similar to the chosen pixel within a square region centered at the chosen pixel. This yields normal vessel pixels in the local region. After extraction, the width of normal vessels is calculated by scanning the width in four directions (horizontal, vertical, two diagonals) and setting the width to the minimum of the four values. A joint histogram of HSV is computed from the detected normal vessel pixels and used as the desired color density for the specified region. Because the colors of most extracted vessel pixels are reddish, this joint density can be further restricted to a sub-region of the original color space.

Generate NV Color. The color of simulated NV pixels is generated one by one by sampling the HSV color distribution of normal vessel segment. Theoretically, this may create inhomogeneity of color on the NV since we do not consider the spatial correlation of the pixels. This is not a practical concern as the density is highly peaked and thus colors are mostly similar. Fig. 6 shows the appearance of simulated NVs with sampled colors.

Blending NV with Background. Fig. 6 shows that the simulated NVs are too salient to be natural due to the clear-cut boundaries. True NVs have the following important appearance characteristics which have not been considered yet: real NVs look well "blended" with surroundings; real NV may become progressively invisible as the vessel extends; also, the color of central vessel pixels are typically more visible than that of the NV boundary pixels; color is not uniform along the branch.

We developed the following strategies to simulate these characteristics:

1. The color of NV pixels (for H, S and V) belonging to the first 70% of one branch is computed as the weighted average of sampling color of this pixel and the color of corresponding pixel underneath, as in Eq. 4, where α is a weight. The larger the weight, the more visible the simulated color with respect to the background color.

$$NV\_Color = \alpha \bullet Sampling\_Color + (1-\alpha) * BG\_Color \tag{4}$$

2. To make the color of the central vessel pixels more visible than that of the NV boundary pixels, we create a bending matrix whose entry corresponds to the weight α that controls how much the sampled NV color contributes to the final color. And more importantly the weights of interior matrix elements decrease linearly to the weights of periphery elements. Note that the color of those background pixels not located on the binary NV structure should not be modified by the blending matrix, so we make the size of the blending matrix be equal to that of the enlargement disk discussed in *caliber enlargement*. A blending matrix with radius 2 is illustrated below. The weights decrease from inner (*maxbld* =0.6) to outer (*minbld* =0.2).

$$\begin{pmatrix} 0 & 0.2 & 0.3333 & 0.2 & 0 \\ 0.2 & 0.4667 & 0.6 & 0.4667 & 0.2 \\ 0.3333 & 0.6 & 0.6 & 0.6 & 0.3333 \\ 0.2 & 0.4667 & 0.6 & 0.4667 & 0.2 \\ 0 & 0.2 & 0.3333 & 0.2 & 0 \end{pmatrix} . \tag{5}$$

3. For the last 30% pixels on a branch, a blending matrix is also created using the above method but with varying *minbld* and *maxbld* for different pixels: they both decrease linearly to 0 as the path reaches its end.

This approach leads to results shown in Fig. 7 where considerable improvement comparing to Fig. 6 can be observed.

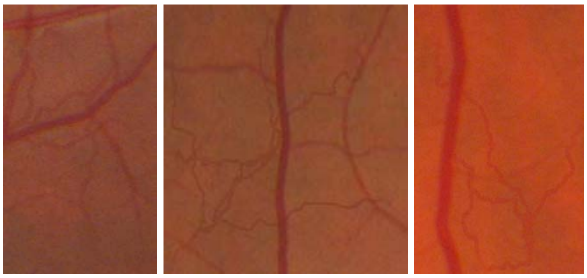
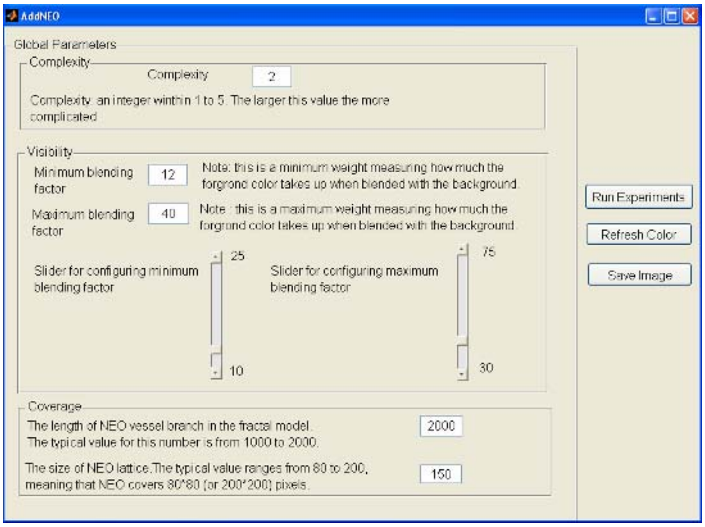


Fig. 7. Neovascularization after color blending

3 An Interactive GUI

If positions to insert simulated NVs were determined by randomly selecting pixels from normal vessels which are automatically detected (e.g., by methods of [15, 16]), our simulation algorithms would have formed a fully automatic system. However, fully automatic approaches have two drawbacks: (1) Point randomly picked on the detected vessel may not be the location preferred by ophthalmologists; (2) More severely, existing vessel detection approaches are not perfect, possibly entailing clinically meaningless results if the point is located on false detections. To address these issues, we propose an interactive approach: normal lesion detection is initialized by an ophthalmologist selecting a point on the vessel (as in *Extract Normal Vessel*). This greatly improves the performance of vessel detection. Moreover, the approach provides instant visual feedback to allow immediately rejection of the unrealistic simulations. Implementing the approach as an interactive GUI also enables the user to adaptively configure many of the algorithmic parameters if the default setting does not generate satisfactory results. A screen shot of the current GUI is shown in Fig. 8.



Global Parameters

Complexity

Complexity: 2

Complexity: an integer winthin 1 to 5. The larger this value the more complicated

Visibility

Minimum blending factor: 12

Note: this is a minimum weight measuring how much the forgrond color takes up when blended with the background.

Maximum blending factor: 40

Note : this is a maximum weight measuring how much the forgrond color takes up when blended with the background.

Slider for configuring minimum blending factor: 25 / 10

Slider for configuring maximum blending factor: 75 / 30

Coverage

The length of NEO vessel branch in the fractal model. The typical value for this number is from 1000 to 2000. 2000

The size of NEO lattice.The typical value ranges from 80 to 200, meaning that NEO covers 80*80 (or 200*200) pixels. 150

Fig. 8. An interactive GUI for NV simulations

There are three parameter panels in the GUI: Complexity, Visibility and Coverage.

Complexity ranges from 1 to 5. This parameter allows a user to create simple to complex NVs. NV may be sinuous and random in shape (*Complexity* = 5) or simple (*Complexity* = 1).

Visibility: Two parameters, *minimum blending factor* and *maximum blending factor*, are associated with visibility. Minimum and maximum blending factors correspond respectively to the minimum and maximum value in the blending matrix.

Coverage: Two parameters, *length of vessel branch* and *size of NV square lattice*, are associated with NV coverage. The length of the NV branch is measured in terms of the number of random walk steps. The size of NV square lattice gives the area that an NV covers.

In a typical run, once all the parameters are set, the user clicks the button "Run Experiments" to start NV simulation for a set of input images. The system reads one background image and displays it on the screen. Then the user selects one or more locations to add NV. Next the system performs the NV simulation algorithm listed in Fig. 1 and displays simulated NVs on the screen. At this time, the user can examine the simulated NV to see if it is realistic. If unsatisfactory, the user can adjust the blending slider to improve the blending effect. The image with satisfying simulated NVs is stored by clicking the "Save Image" button. This can be repeated for different background images.

It must be noted that, although the system is interactive, a large portion of the process is automatic, and ideally, the user will only need to select the points without further adjusting. Thus a performance factor of the system is the acceptance rate, i.e., the percentage of the simulated NVs that do not need further adjustment.

4 Experiments and Evaluation

We have tested the system on natural human retina images. Since NV occurs during the stage of proliferative DR, some non-proliferative DR lesions such as microaneurysm or hemorrhages are usually present at that time. So images with non-proliferative lesions are selected from a database to serve as the background images into which the simulated NVs are inserted. Some simulated NVs are shown in Fig. 9, where background images with different pigments are used, illustrating that the proposed approach is able to adapt to the appearance of background images. In Fig. 10, images with simulated NVs are intentionally mixed with real NV images, illustrating that the system is able to generate NVs that are indistinguishable from the real ones. Since the image of the entire retina is too large (1024*768), only the area of retina with simulated NV is shown here.

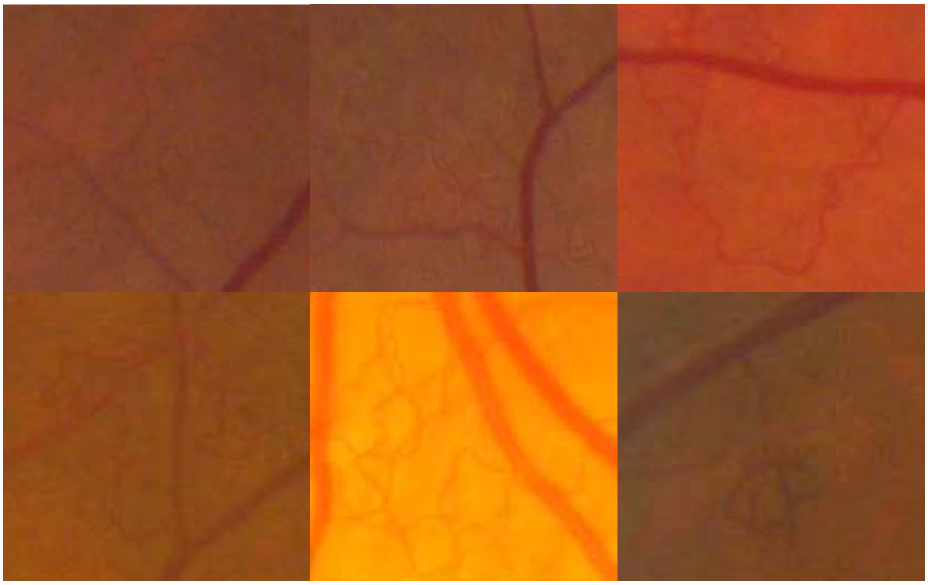


Fig. 9. Examples of photorealistic simulated NV. Here, the width of the simulated vessel varies along the path. Color also presents various degree of saturation.

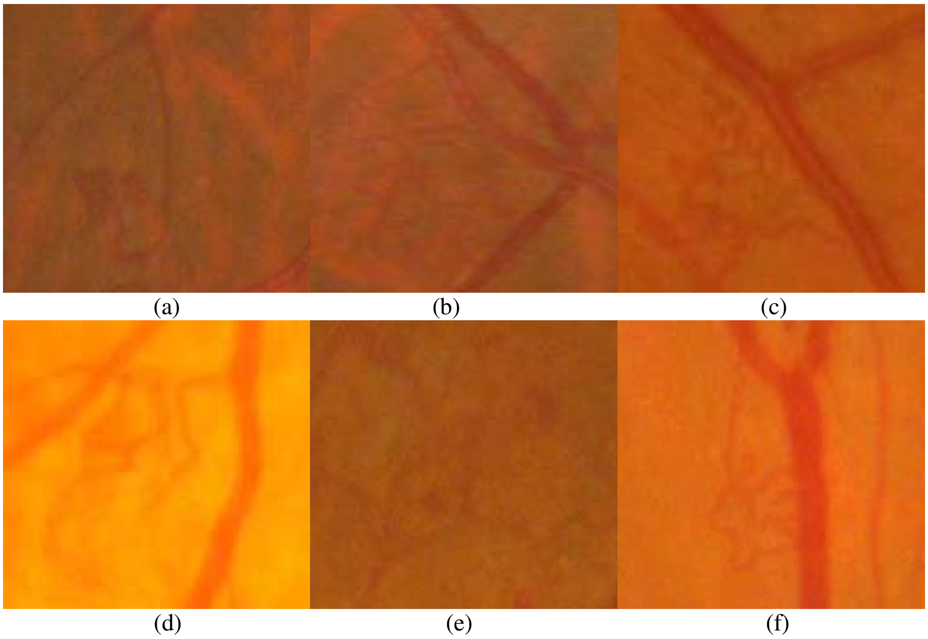


Fig. 10. Images with either simulated or real NVs (but not both). Would you be able to tell the difference? See[1] for answer.

The complete system is under deployment for ophthalmologists' formal evaluation of its performance, including the acceptance rate, in a clinic setting. The initial results suggest that this is a promising method.

5 Conclusion and Future Work

In this paper we present a diabetic retinopathy neovascularization simulation system. The shape of NV is generated by self-avoiding random walk and self-intersecting random walk (spreading percolation and invasion percolation have also been tested, but not discussed here). The color of NV is generated by sampling the color distribution of normal retina vessel. In addition we have developed an interactive system to provide user feedback for optimal performance of the automated algorithms. Experiments on images with non-proliferative DR lesions show that the system is able to simulate NVs indistinguishable to the real ones.

There exist known limitations in the current system, i.e., it cannot create some particular NV patterns such as flower-like NVs, which will be addressed in our future work. Another challenging problem is, currently the performance of the system relies on subjective judgment of ophthalmologists, and thus it is difficult to obtain statistics from a large pool of ophthalmologists to study possible bias/variance of the subjects.

[1] In Fig. 10, (a), (d) and (e) are real NVs, others are simulated.

It is likely that the fractal dimension might be the quantitative index of measuring the similarity between simulated NVs and real NVs, which is one possible working direction for our future research.

Acknowledgment

This project was supported by grant #1-2002-212 from the Juvenile Diabetes Research Foundation International.

References

1. Javitt, J. C., Aiello, L. P., Chiang, Y., Ferris, F. L., Canner, J. K. S., Greenfield: Preventive eye care in people with diabetes is cost saving to the federal government. Diabetes Care. 17(8): 909-917 (1994)
2. ETDRS Research Group: Early photocoagulation for diabetic retinopathy: Early treatment diabetic retinopathy study report number 9. Ophthalmology. (1998) 766–785
3. Lee, S., *et al*: Comparison of diagnosis of early retinal lesions of diabetic retinopathy between a computer and human experts. Arch Ophthalmol. 119: 509–515 (2001)
4. Usher, D., *et al*: Automated detection of diabetic retinopathy in digital retinal images: a tool for diabetic retinopathy screening. Diabet Med. Jan. 21(1):84-90 (2004)
5. Hipwell, J.H., *et al*: Automated detection of microaneurysms in digital red-free photographs: a diabetic retinopathy screening tool. Diabet Med. 17(8):588-94 (2000)
6. Sinthanayothin, C., *et al*: Automated detection of diabetic retinopathy on digital fundus images. Diabet. Med. 19(2):105-12 (2002)
7. Teng, T., *et al*: Progress towards automated diabetic ocular screening: a review of image analysis and intelligent systems for diabetic retinopathy. Medical & Biological Engineering & Computing. 40(1):2-13 (2002)
8. Early Treatment Diabetic Retinopathy Study Research Group: Grading diabetic retinopathy from stereoscopic color fundus photographs: An extension of the modified Airlie House classification. ETDRS Report Number 10. Ophthalmology 98: 786-806 (1991)
9. Landini, G., Misson, G.: Simulation of corneal neovascularization by inverted diffusion limited aggregation. Invest Ophthalmol Vis Sci. 34(5):1872-5 (1993)
10. Hoe, C.L., Samei, E., Frush, D.P., Delong, D.M.: Simulation of liver lesions for pediatric CT. Radiology. 238(2):699-705 (2006)
11. Cross, S. S.: Fractal in pathology. Journal of Pathology. 182**:**1–8 (1997)
12. Mandelbrot, B.: *The Fractal Geometry of Nature*. San Francisco: WHFreeman. (1982) 460
13. Masters, B. R.: Fractal analysis of the vascular tree in the human retina. Annu. Rev. Biomed. Eng. 6:427–52 (2004)
14. Vicsek, T.: Fractal Growth Phenomena. World Scientific Pub Co Inc, 2nd edn. (1992) 105-111 and 111-114
15. Can, A., Stewart, C.V., Roysam, B., and Tanenbaum, H.L.: A Feature-based, robust, hierarchical algorithm for registering pairs of images of the curved human retina. IEEE Transactions on PAMI. 24:347-364 (2002)
16. Lalibert, F., Gagnon, L., and Sheng, Y.: Registration and Fusion of Retinal Images-An Evaluation Study. IEEE Transactions on Medical Imaging. 22:661–673 (2003)

17. Patton, N., Aslam, T.M., *et al*: Retinal image analysis: concepts, applications and potential. Progress in Retinal and Eye Research. 25(1):99-127 (2006)
18. Stanley, H.E., Amaral, L.A.N., Buldyrev, S.V., Goldberger, A.L., Havlin, S., *et al*: Scaling and universality in living systems. *Fractals. 4:427-51 (*1996)
19. Masters, B., Platt, D.: Development of human retinal vessels: a fractal analysis. Invest. Ophthalmol. Vis. Sci. 30 (Suppl.):391 (1989)
20. Family, F., Masters, B., Platt, D.: Fractal pattern formation in human retinal vessels. Physica D 38:98-103 (1989)
21. Vicsek, T.: Fractal Growth Phenomena. World Scientific Pub Co Inc, 2nd edn. (1992) 119-135

Mesh Optimisation Using Edge Information in Feature-Based Surface Reconstruction

Jun Liu and Roger Hubbold

Department of Computer Science, University of Manchester
Manchester, M13 9PL, United Kingdom
{jun, roger}@cs.man.ac.uk

Abstract. One of the most challenging and fundamental problems in computer vision is to reconstruct a surface model given a set of uncalibrated 2D images. Well-established Structure from Motion (SfM) algorithms often result in a sparse set of 3D surface points, but surface modelling based on sparse 3D points is not easy. In this paper, we present a new method to refine and optimise surface meshes using edge information in the 2D images. We design a meshing – edge point detection – re-meshing scheme that can gradually refine the surface mesh until it best fits the true physical surface of the object being modelled. Our method is tested on real images and satisfactory results are obtained.

1 Introduction

Much attention has been paid to the task of obtaining a surface representation from 3D data points on an unknown surface. Early work has been focused on the data acquired with a laser range scanner, with the characteristics that the obtained 3D point cloud is dense and well-distributed [1, 2, 3, 4]. Although these methods have been reported to be successful, the nature of the range scanning technique greatly limits its usefulness in real-world applications: it is an "invasive" technique in that the ray emitted by the scanner may damage the object being scanned; the scanning device is often very expensive; the scanning process is very slow even for a moderately sized object, and thus not suitable for modelling large scenes such as buildings.

The advances in computer vision technologies provide an exciting alternative for surface reconstruction. One of the most challenging and fundamental problems in computer vision is reconstruction of a surface model given a set of uncalibrated 2D images captured by a hand-held camera. Although this is not a solved problem, progress has been made in the last decade and a few working systems have been built. The first steps usually involve Structure from Motion (SfM) [5] and camera auto-calibration [6], delivering camera pose information as well as a sparse 3D point reconstruction based on image feature points. Pollefeys et al. [7] then applied dense stereo matching techniques on the images which results in a per-pixel density reconstruction of the scene. Lhuillier and Quan [8, 9] adopted a different approach by propagating points on the images to obtain a "quasi-dense" reconstruction.

Even with “dense” stereo matching, the 3D data from multiple images are sparse, noisy, and irregularly distributed compared to those from range scanners, therefore traditional 3D surface reconstruction techniques used with range data cannot be used with the data from passive computer vision systems. Fortunately, the extra 2D image information is still available which can be used to facilitate surface reconstruction. Lhuillier and Quan [8] proposed a variational approach integrating 3D stereo data with 2D image information; Solem and Heyden [10] addressed the problem based on methods for region tracking on surfaces and moving implicit curves; Paris et al. [11] used global graph cut optimisation to find optimal surface patches.

An interesting divergence from dense-stereo-data-based surface reconstruction is “feature-based surface reconstruction” [12, 13]: the sparse 3D points obtained with SfM techniques are directly fed into the surface reconstruction algorithm. This is based on the observation that the human vision system tends to recognise objects by salient features such as edges and points. Feature-based surface reconstruction has the advantage that it represents the scene models in a more efficient way: meshes are greatly simplified, thus the cost of computation and visualisation is greatly reduced. It also has the advantage over dense-stereo-based methods in that feature based surface reconstruction allows for much wider base lines in the input images, which often prevents the dense stereo matching algorithms from working. However, finding meshes that correspond to the true shape of scenes being modelled based on sparse 3D points is very difficult. This is probably the reason why feature-based surface reconstruction remains largely unpopular.

This paper addresses the mesh optimisation problem in feature-based surface reconstruction. The remainder of the paper is organised as follows: Section 2 discusses the limitations of previous work and briefly states the advantages of our method; Section 3 formally specifies the problem; Section 4 presents a robust method to detect points lying on the image edges; Section 5 presents a method to search for the correct 3D positions for the identified edge points; Section 6 shows some experimental results and Section 7 concludes our work with suggestions for future work.

2 Related Work

The problem of mesh optimisation in feature-based surface reconstruction was first addressed by Morris and Kanade [14]. In their pioneering work an initial mesh based on Delaunay triangulation is first obtained. An edge swapping technique is then applied to traverse possible topologies of the 3D feature points. They use a greedy algorithm to search for the best triangulation corresponding to the most consistent topology across multiple views. Vogiatzis et al. [15] extended this work by using simulated annealing instead of a greedy algorithm to search for the best triangulation, making the optimisation less susceptible to local minima. Most recently Nakatuji et al. [16] detect texture discontinuities in the images and swap the edges to minimise the overall discontinuity of the triangulation.

In general, previous methods suffer from several problems:

1. They are heavily dependent on a judicious selection of feature points on edges and corners. If no feature points are found (or cannot be reliably tracked) on edges and corners, then edge swapping will not return a triangulation that reflects the true surface of the object.
2. Finding the optimal triangulation by edge swapping can easily get stuck in local minima. Statistical methods such as simulated annealing are computationally expensive, and their convergence cannot be guaranteed. Therefore, the previous methods can only work with a small set of 3D points.

Our Contribution. We present a new method for mesh optimisation in feature-based surface reconstruction by directly incorporating edge information in the triangulation process. Our method is different from previous work in that we do not rely on an edge swapping technique, whose complexity is exponential to the number of input 3D points. Instead, we adopt a meshing – edge point detection – re-meshing scheme. Our method has several advantages over the previous methods:

1. No edge swapping is involved. We only rely on the well-established 2D Delaunay triangulation algorithm. Our algorithm is readily extensible to large sets of 3D feature points.
2. Contrary to previous methods, triangle splitting is permitted (and is essential) in our method, allowing for a more general distribution of features on the object surface, i.e. features do not need to lie strictly on the corners.
3. Our method can be iterated multiple times to further refine the quality of the resultant surface mesh.

3 Problem Statement

Given a set of 3D points lying on the surface of an object, there exist many possible surface triangulations passing through all these points. A simple example is shown in Figure 1. Although both triangulations in Figure 1 are valid configurations in 2D as well as in 3D, only triangulation (a) is consistent with the true object surface. In practice we do not know the true surface but instead have a set of images of the object. The goal is to resolve this ambiguity by selecting a triangulation based on its consistency with this set of images of the object.

The problem can be mathematically formulated as follows: we have as input a set of n images $\mathbf{I} = \{\mathtt{I}_1, ..., \mathtt{I}_n\}$, a set of m 3D points $\mathbf{X} = \{\mathtt{X}_1, ..., \mathtt{X}_m\}$ obtained with a SfM algorithm, and n projection matrices $\mathbf{P} = \{\mathtt{P}_1, ..., \mathtt{P}_n\}$ which define the transformation from 3D points to 2D points for each image. We define our triangular mesh model as $\mathbf{M} = \{\mathbf{V}, \mathbf{E}\}$ where $\mathbf{V}$ is a set of 3D points and $\mathbf{E}$ is a set of edges connecting members of $\mathbf{V}$. Our goal is to find a mesh model $\mathbf{M}$ that maximises the conditional likelihood Pr

$$\underset{\mathbf{M}}{\operatorname{argmax}} \Pr(\mathbf{M} \mid \mathbf{I}, \mathbf{X}, \mathbf{P}) \tag{1}$$

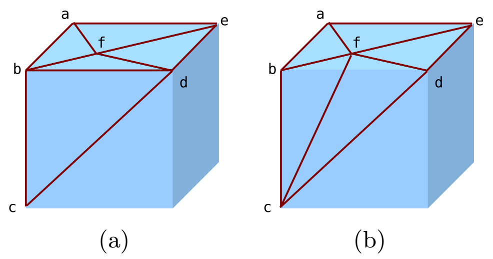


Fig. 1. Two of the many possible triangulation from 3D points on a cube. (a) Triangulation that corresponds to the true physical surface; (b) Triangulation that does not corresponds to the true physical surface: points from different planar surfaces are triangulated.

Note that **V** and **X** are different in our formulation. Unlike previous methods [14, 15, 16], where no new 3D points are added and only edges are swapped, we add new 3D points in **X** to refine the mesh topology. Hence, **X** is a subset of **V** in our formulation.

4 Edge Point Detection

One of the methods to create an initial mesh model is to perform a Delaunay triangulation on the 2D feature points from one of the images and project it into 3D space [17, 7]. This method works well for dense stereo reconstruction but for feature-based surface reconstruction, it often triangulates points from different planar surfaces (as shown in Figure 1(b)), leading to artifacts when the object is viewed from different angles.

Since edges in the images are natural indicators of surface discontinuities, it is advisable to include edge information in the triangulation process. However, points on edges are difficult to track across the images and hence many feature detection algorithms such as SIFT [18, 19] deliberately discard points lying on edges. Furthermore, traditional edge detectors such as the Canny edge detector [20] and other gradient-based techniques, although successful in many application areas, tend to give false positive responses in the presence of highly textured objects. All these factors pose difficulties in applying the edge information in guiding the triangulation process.

Fortunately, in this particular problem, we are only interested in finding the *intersection* of images edge with edges in mesh triangles rather than the *integral* edges as a whole. This observation leads us to designing a specific "edge point detection" algorithm.

4.1 Problem Re-formulation

Consider that after Delaunay triangulation on the feature points on a reference image, as shown in Figure 2(a), the goal is to find a point e on the triangle edge

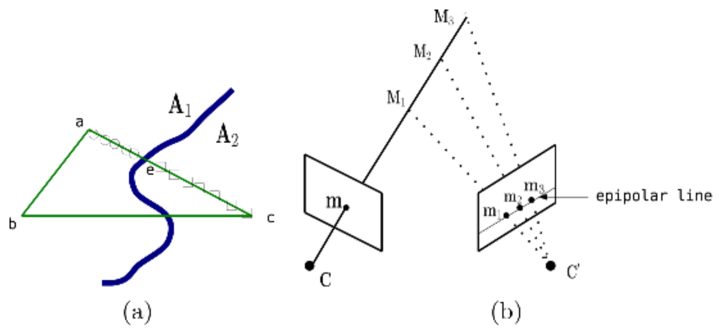


Fig. 2. (a) Naive Delaunay triangulation inevitably connects points from different areas (areas $\mathbf{A}_1$ and $\mathbf{A}_2$) together. The goal is to find the intersection point e lying on the discontinuity along the edge $a \rightarrow c$. (b) Multiple view geometry. The 3D point corresponding to image point m lies in the ray passing through m from the image centre $\mathbf{C}$. When only one view is available, there is not enough information to determine the 3D location: any point lying on the ray is a possible candidate. However, when more views are available, the location can be identified by computing the reprojection on another image by camera $\mathbf{C}'$: if $\mathbf{M}_k$ is the correct 3D point, then its reprojected 2D point $\mathbf{m}_k$ should have similar appearance with $\mathbf{m}$.

$a \rightarrow c$ that lies on the discontinuity (image edge separating areas $\mathbf{A}_1$ and $\mathbf{A}_2$). If we travel along the edge $a \rightarrow c$ and record the pixels in a vector $\mathbf{S} = \{\mathbf{p}_1, ..., \mathbf{p}_n\}$, then we need to find a pixel $\mathbf{p}_k$ where $1 < k < n$ such that

$$\operatorname*{argmax}_k \Pr(\mathbf{S}_1^{k-1} \mid \mathbf{A}_1) \Pr(\mathbf{S}_{k+1}^{n} \mid \mathbf{A}_2) \tag{2}$$

where $\mathbf{S}_1^{k-1} = \{\mathbf{p}_1, ..., \mathbf{p}_{k-1}\}$ and $\mathbf{S}_{k+1}^{n} = \{\mathbf{p}_{k+1}, ..., \mathbf{p}_n\}$.

4.2 Maximum-Likelihood Estimation and KL Divergence

In practice the distribution models of $\mathbf{A}_1$ and $\mathbf{A}_2$ are not known a priori unless some texture segmentation techniques are used. Therefore the formulation in Equation 2 is not readily applicable to our problem. However, we do know a priori that $\mathbf{A}_1$ and $\mathbf{A}_2$ are different from each other. If we can define an appropriate distance function $D(\mathbf{S}_i, \mathbf{S}_j)$ to measure the (dis-)similarity of the two segments $\mathbf{S}_i$ and $\mathbf{S}_j$, then we can formulate our problem in a maximum-likelihood estimation (MLE) framework

$$\operatorname*{argmax}_k D(\mathbf{S}_1^{k-1}, \mathbf{S}_{k+1}^{n}) \tag{3}$$

In other words, we want to find a k such that $\mathbf{S}_1^{k-1}$ and $\mathbf{S}_{k+1}^{n}$ are most different from each other.

It is safe to assume that the pixel set sampled from area $\mathbf{A}$ follows a Gaussian distribution N with mean μ and standard deviation σ. If the pixels are sampled from different areas, then they follow a mixture of Gaussian distribution $\mathcal{N}$ with mean μ and covariance matrix Σ. Kullback-Leibler (KL) divergence can be used to measure the cross-entropy (i.e. dis-similarity) between two Gaussian mixtures f and g:

$$D_{\mathrm{KL}}(f \parallel g) = \int f \ln \frac{f}{g} \tag{4}$$

where $f \sim \mathcal{N}(\mu_f, \Sigma_f)$ and $g \sim \mathcal{N}(\mu_g, \Sigma_g)$.

Since there is no closed-form expression for KL-divergence between two mixture of Gaussians, computing this distance measure is usually done using Monte-Carlo simulation, which causes a significant increase in computational complexity. An approximation is proposed by Goldberger et al. [21] which leads to a closed-form solution:

$$D_{\mathrm{KL}}(f \parallel g) = \frac{1}{2}\left(\ln \frac{|\Sigma_f|}{|\Sigma_g|} - d + tr(\Sigma_f^{-1}\Sigma_g) + (\mu_g - \mu_f)^{\mathsf{T}}\Sigma_f^{-1}(\mu_g - \mu_f)\right) \tag{5}$$

where d is the dimensionality of the Gaussian mixtures. In our problem $d = 3$ because we model the RGB plane of each image pixel separately.

4.3 Algorithm Description

It is sensible to combine gradient-based edge detection with maximum-likelihood estimation. For each edge in the triangle, we travel through the pixels and find intensity discontinuities by computing the first-order derivative in the travelling direction. Suppose we record n pixels in one triangle edge $\mathbf{S} = \{\mathbf{p}_1, ..., \mathbf{p}_n\}$, a good approximation to measure discontinuity d for pixel $\mathbf{p}_k$ is

$$d(\mathbf{p}_k) = |\mathbf{p}_k - \mathbf{p}_{k-1}| + |\mathbf{p}_k - \mathbf{p}_{k+1}| \tag{6}$$

Gradient-based edge detection doesn't work well for highly textured areas. Imagine that in Figure 2(a), $\mathbf{A}_1$ is highly textured and has repetitive pattern while $\mathbf{A}_2$ is texture-less, then gradient-based edge detection will return many positive responses in $\mathbf{A}_1$, which is not desirable for finding true discontinuities. In this case, maximum-likelihood estimation can be used to discard false positive responses: the positive responses from gradient-based detection can be used as candidates in maximum-likelihood estimation framework, and the true discontinuity corresponds to the pixel $\mathbf{p}_k$ that maximises the KL-divergence D_{KL} between $\mathbf{S}_1^{k-1}$ and $\mathbf{S}_{k+1}^{n}$.

5 3D Position Identification

The detected edge points described in Section 4 do not provide any extra information to facilitate surface reconstruction unless their corresponding positions in 3D space are identified. Estimating 3D positions of feature points lying on edges in the SfM stage is not easy, as points along the edge usually have similar appearance, and hence they are difficult to identify and track across the images. However, camera poses and sparse reconstructed features are available after SfM.

This information can help constrain the search-space of the 3D positions of the detected 2D edge points.

Ideally, the corresponding 3D point of a 2D image point lies on the back-projected ray from the camera centre passing through the 2D image point (see Figure 2(b)). If we can identify the true location of the 3D point, then its reprojected point in each image should have similar appearances as they correspond to the same feature. Therefore, the 3D point identification problem can be viewed as an inverse problem of SfM.

In practice, however, inevitably there will be noise in SfM and camera auto-calibration processes. Therefore, the position of 3D point may show slight deviation from the ray. We propose to propagate the search-space such that it can take noise into account, as shown in Figure 3. $\mathbf{m}_1$ and $\mathbf{m}_2$ are two vertices of a triangle (obtained from Delaunay triangulation on the feature points) and $\mathbf{M}_1$ and $\mathbf{M}_2$ are the corresponding 3D points respectively (as after SfM their 3D locations are already known). An edge point $\mathbf{e}$ is detected with the method described in Section 4. The search-space for its 3D location is a uniformly sampled 3D grid centred at $\mathbf{E}$. $\mathbf{E}$ is the mid-point of projections of $\mathbf{M}_1$ and $\mathbf{M}_2$ on the viewing ray l which passes through $\mathbf{e}$ from $\mathbf{C}$. The 3D grid is positioned in such a way that it has larger search-space along the ray l and smaller search-space in the direction perpendicular to l. Note that when the search-space perpendicular to l is zero, it reduces to search-space along the ray, ignoring the deviation caused by noise.

We evaluate each 3D point sample $\mathbf{E}_k$ from the grid by computing its reprojected 2D point $\mathbf{e}_i$ on each image $\mathtt{I}_i$ and compare their similarity to the edge point $\mathbf{e}$. If $\mathbf{E}_k$ is the correct 3D point, then its projection in other images should

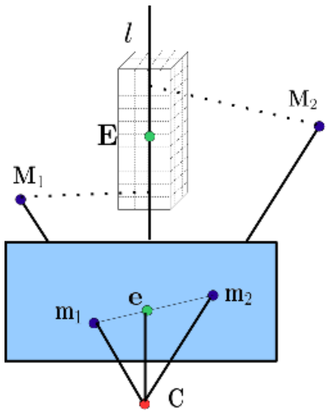


Fig. 3. Search-space propagation. $\mathbf{m}_1$ and $\mathbf{m}_2$ are two vertices of a triangle and $\mathbf{M}_1$ and $\mathbf{M}_2$ are the corresponding 3D points respectively. An edge point $\mathbf{e}$ is detected. The search-space for its 3D location is a uniformly sampled 3D grid centred at $\mathbf{E}$. $\mathbf{E}$ is the mid-point of projections of $\mathbf{M}_1$ and $\mathbf{M}_2$ on the viewing ray l which passes through $\mathbf{e}$ from $\mathbf{C}$. The 3D grid is positioned in such a way that it has larger search-space along the ray l and smaller search-space in the direction perpendicular to l.

have similar appearance to $\mathbf{e}$. The problem can be formulated as finding an $\mathbf{E}_k$ that minimises the cost function

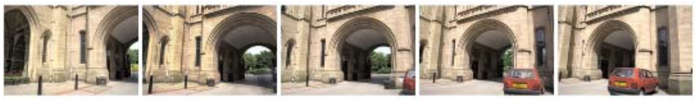
$$\min_{\mathbf{E}_k} \sum_i \mathcal{C}(\mathbf{e}, \mathbf{e}_i) \tag{7}$$

where $\mathcal{C}(\mathbf{e}, \mathbf{e}_i)$ is the cost function measuring the dis-similarity between $\mathbf{e}$ and $\mathbf{e}_i$. Two common criteria are Sum of the Squared Difference (SSD) and Normalised Cross-Correlation (NCC) between pixels in a small window centred at $\mathbf{e}_i$ in each image. In our experiment we use NCC because it is less sensitive to illumination change between the views. We select the window size to be 7×7, as a trade-off between performance and speed.

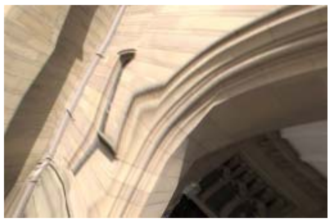
One problem still remains. The visibility property $\mathbf{V}_{\mathbf{E}_k}$ of the 3D point $\mathbf{E}_k$ is not known beforehand: we have no knowledge about in which images the 3D point $\mathbf{E}_k$ is seen. Fortunately, the neighbouring feature points can provide a reasonable approximation for $\mathbf{V}_{\mathbf{E}_k}$. As is shown in Figure 3, if $\mathbf{e}$ is closer to $\mathbf{m}_1$ than to $\mathbf{m}_2$, then we assign the visibility property $\mathbf{V}_{\mathbf{M}_1}$ to $\mathbf{E}_k$; otherwise we assign the visibility property $\mathbf{V}_{\mathbf{M}_2}$ to $\mathbf{E}_k$.

6 Experimental Results

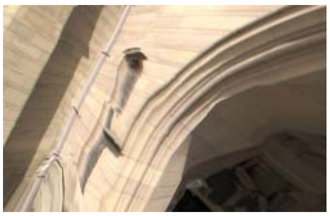
We begin by performing a Delaunay triangulation on the feature points of a reference image. Edge points are detected as described in Section 4 and their 3D locations are identified as described in Section 5. The surface mesh can be optimised by re-meshing the new feature set on the image. Note that this process can be iterated multiple times until no triangle edge cuts through an image edge.



(a)



(b)



(c)

Fig. 4. Test case 1: Arch sequence. (a) 5 of the input images; (b) Close-up view of the original surface model from a very different angle from where the input images are captured. Notice that the artifacts in the area of arch are caused by poor meshing; (c) Enhanced surface model with our method. The curved surface in the arch is correctly modelled.

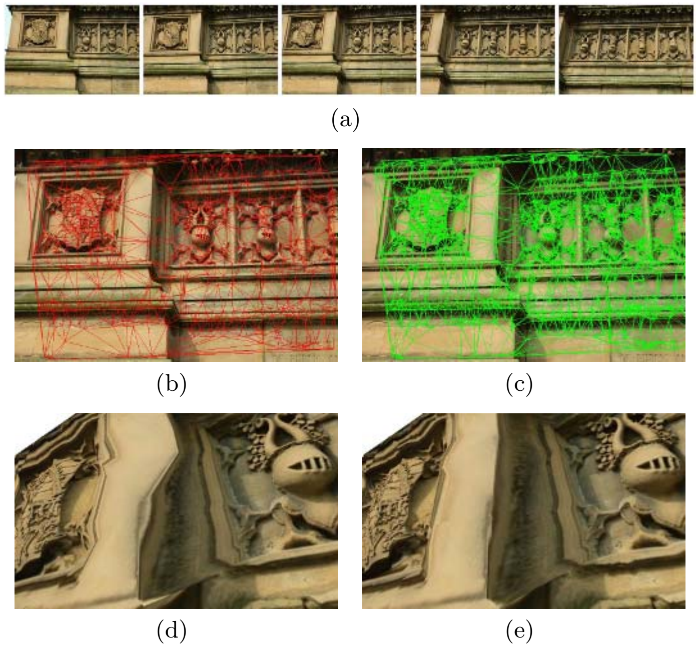


Fig. 5. Test case 2 : Monument sequence. (a) 5 of the input images; (b) Original mesh super-imposed onto the reference image: notice that triangles cut through image edges; (c) Refined mesh, triangulation is well-conditioned and very few triangles cut through image edges; (d) Original surface model viewed from very different angle from where the image is captured: the artifacts along the edge are caused by triangular meshes connecting points from different surfaces; (e) Refined surface model: edge points are detected and their 3D locations are correctly determined. The surface model refined by our method is very consistent with the true physical surface of the object.

3D feature points and camera information are obtained based on our previous work [13]. The reconstruction process is fully automatic and requires no information other than the images alone. Figure 4 shows the reconstructed surface model of an arch. Notice that the fine details of the curved surface are correctly reconstructed, which would be impossible if we use edge swapping techniques. Figure 5 shows another example demonstrating the result of our method. Our method is very efficient: both test cases involve detection and 3D position identification of around 10,000 edge points and it finishes within 10 seconds on a 2GHz processor.

7 Conclusion and Future Work

We presented a new method for mesh optimisation in feature-based surface reconstruction by directly incorporating edge information in the triangulation process. Our method is different from previous ones in that we do not rely on an edge swapping technique. Instead, we adopt a meshing – edge point detection – re-meshing scheme. Experiments on real images show satisfactory results.

Our method still has limitations: it assumes that edges do not change across the images. Although it is true for most occasions, it may not hold when the object has a smoothly curved surface, in which case the edges change according to the viewpoint. Moreover, our method works less well when the edges lie on the epipolar line (See Figure 2(b)): re-projected 2D points are more difficult to distinguish and hence the identified 3D positions are less reliable. In our future work, we plan to solve the above problems by combining our method with the edge swapping method and use a more robust descriptor (such as a SIFT-like descriptor [18, 19]) rather than NCC to match reprojected 2D points.

References

[1] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W.: Surface reconstruction from unorganized points. In: SIGGRAPH. (1992) 71–78

[2] Szeliski, R., Tonnesen, D., Terzopoulos, D.: Modeling surfaces of arbitrary topology with dynamic particles. In: CVPR. (1993) 82–87

[3] Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: SIGGRAPH. (1996) 303–312

[4] Zhao, H., Osher, S., Merriman, B., Kang, M.: Implicit and nonparametric shape reconstruction from unorganized data using a variational level set method. CVIU **80**(3) (2000) 295–314

[5] Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. Second edn. Cambridge University Press (2003)

[6] Triggs, B.: Autocalibration and the absolute quadric. In: CVPR. (1997) 609

[7] Pollefeys, M., Gool, L.V., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., Koch, R.: Visual modeling with a hand-held camera. IJCV **59**(3) (2004) 207–232

[8] Lhuillier, M., Quan, L.: Surface reconstruction by integrating 3D and 2D data of multiple views. In: ICCV. Volume 02. (2003) 1313

[9] Lhuillier, M., Quan, L.: A quasi-dense approach to surface reconstruction from uncalibrated images. TPAMI **27**(3) (2005) 418–433

[10] Solem, J.E., Heyden, A.: Reconstructing open surfaces from unorganized data points. In: CVPR. Volume 02. (2004) 653–660

[11] Paris, S., Sillion, F., Quan, L.: A surface reconstruction method using global graph cut optimization. IJCV **66**(2) (2006) 141–161

[12] Taylor, C.J.: Surface reconstruction from feature based stereo. In: ICCV. Volume 01. (2003) 184

[13] Liu, J., Hubbold, R.: Automatic camera calibration and scene reconstruction with scale-invariant features. In: ISVC. (2006) to appear

[14] Morris, D., Kanade, T.: Image-consistent surface triangulation. In: CVPR. Volume 1. (2000) 332–338

[15] Vogiatzis, G., Torr, P., Cipolla, R.: Bayesian stochastic mesh optimisation for 3D reconstruction. In: BMVC. Volume 2. (2003) 711–718

[16] Nakatuji, A., Sugaya, Y., Kanatani, K.: Mesh optimization using an inconsistency detection template. In: ICCV. (2005) 1148–1153

[17] Pollefeys, M., Koch, R., Gool, L.V.: Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In: ICCV. (1998) 90

[18] Lowe, D.G.: Object recognition from local scale-invariant features. In: ICCV. (1999) 1150
[19] Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV **60**(2) (2004) 91–110
[20] Canny, J.: A computational approach to edge detection. TPAMI **8** (1986) 679–714
[21] Goldberger, J., Gordon, S., Greenspan, H.: An efficient image similarity measure based on approximations of KL-divergence between two Gaussian mixtures. In: ICCV. Volume 1. (2003) 487

Finite Sample Bias of Robust Scale Estimators in Computer Vision Problems

Reza Hoseinnezhad[1], Alireza Bab-Hadiashar[1], and David Suter[2]

[1] Swinburne University of Technology, Australia
[2] Monash University, Australia

Abstract. In computer vision applications of robust estimation techniques, it is usually assumed that a large number of data samples are available. As a result, the finite sample bias of estimation processes has been overlooked. This is despite the fact that many asymptotically unbiased estimators have substantial bias in cases where a moderate number of data samples are available. Such cases are frequently encountered in computer vision practice, therefore, it is important to choose the right estimator for a given task by virtue of knowing its finite sample bias. This paper investigates the finite sample bias of robust scale estimation and analyses the finite sample performance of three modern robust scale estimators (Modified Statistical Scale Estimator, Residual Consensus estimator and Two-Step Scale Estimator) that have been used in computer vision applications. Simulations and real data experiments are used to verify the results.

1 Introduction

Robust estimation techniques have been extensively used in computer vision applications including optic flow computation [1], structure from motion, range and motion segmentation, etc [2]. In those applications, it is often assumed (sometimes implicitly) that a large number of data samples are available for the estimation process. Scrutiny of common computer vision applications reveals that this assumption is not always justified. For instance, in the well-known problem of structure from motion, corresponding features of various objects seen by a pair of cameras are used to calculate the fundamental matrix [3]. In a typical scene, moving objects would only have a small number of matching features and therefore, the parameter estimates will be significantly affected by the finite sample bias of the applied estimators.

A similar problem arises in many parametric range segmentation schemes. As robust estimators are increasingly used to segment objects with a small number of data points [4,5,6], the finite sample bias of the applied estimators is of significance. An example is shown in Fig. 1 where the door of a lift is scanned by a SICK laser range scanner for an indoor map building exercise. The scanner device produces 201 data samples (0-100° with the resolution 0.5°) in each scan. When the door is scanned from a distance, a small number of the 201 samples represent the door of the lift. Figures 2(a) and 2(b) show the measured range data in two scenarios: In Fig. 2(a), the door is near and more than 90 range data samples belong to the lift door. The robust estimators can easily distinguish the door from its side walls. In Fig. 2(b), the door is far from the measuring device and a small number (around 20) of data samples belong to the door. In this case, the

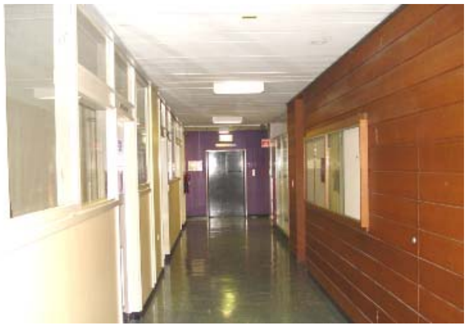
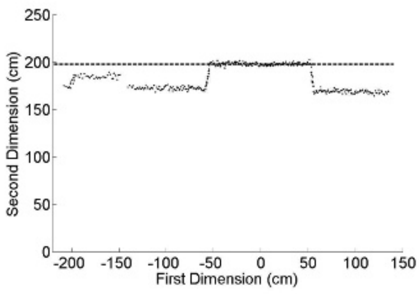
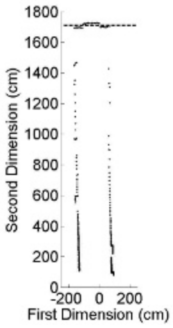


Fig. 1. A lift door to be modelled in a range segmentation application



(a)



(b)

Fig. 2. Range scan from (a) short distance (b) far distance

bias of the estimators is too large to distinguish the target structure from its surrounding walls. As a result, the final map wrongly indicates that the front wall is a planar surface.
Stewart's work [7] appears to be the only attempt in the computer vision literature to analyse the bias of robust estimators. That analysis only includes *asymptotic* bias of the estimators and does not investigate the finite sample bias.

It is important to note that an asymptotically unbiased (consistent) estimator is not necessarily unbiased if it is applied to a finite number of samples. Thus, in practice, one needs to know the minimum number of data samples that guarantees the robust estimator to be unbiased (or almost unbiased). Such a knowledge is required to choose an appropriate estimator for a particular problem or to find ways to correct the finite sample bias of a preferred estimator.

Modern high breakdown robust estimators [6, 4, 5] include a robust scale estimator at their core that calculates the noise scale for any given fit. The best fit is then found by minimising a cost function that is either the noise scale itself, or a function of the estimated noise scale. Scale estimation is therefore the critical part of such estimators and if the scale is wrongly calculated, a proper fit will not be found, particularly in scenarios involving multiple close structures. Thus, the performance of these estimators largely depends on their scale estimation part. This paper investigates the finite sample bias of robust scale estimators developed to solve computer vision applications. As such, we study the limitations incurred by: 1) finiteness of the number of samples 2) the mutual distances of multiple structures in a given data set.

2 Finite Sample Bias of Scale Estimators: Definition

Before presenting our approach, some notations and assumptions are introduced. The residuals of a hypothesised fit θ^* (the closest fit to the true fit θ that can be found through a search scheme like random sampling) are the only data samples utilised by a scale estimator to calculate the scale. In practice, the residuals are a finite ensemble of a statistical population whose characteristics mainly depend on the distribution of the data points (denoted by H). For example if there is only one structure in the data with no outliers, then the residuals are usually assumed to have a normal distribution with its standard deviation equal to the noise scale (yet to be estimated). The probability density function (pdf) of the absolute residuals are denoted by $f^a(r|\theta^*, H)$. More information on calculation of this function from data distributions can be found in [7]. In this paper, we assume that the hypothesised fit is close to the true fit and the residuals are independently and identically distributed (iid). Thus, the joint pdf of all absolute residuals is equal to $\prod_{i=1}^{n} f^a(r_i|\theta^*, H)$ where n is the total number of data samples. We define the following normalised measure for the finite sample bias of a scale estimator:

Definition 1. *If σ is the true scale and $E[\hat{\sigma}_n^2|\theta^*, H]$ is the statistical mean of an estimated scale for a given hypothesised fit θ^* and a specific data distribution H, then the bias of the estimator is:*

$$\xi(n, \theta^*, H) \triangleq 100\% \times \left|E[\hat{\sigma}_n^2|\theta^*, H] - \sigma^2\right| / \sigma^2. \tag{1}$$

The arguments n, θ^* and H emphasise that the bias depends on the number of data samples, the hypothesised fit and the data population, but not on the scale itself. Indeed, the above is a scale invariant definition of bias. To calculate the bias of a scale estimator as defined in equation (1), the statistical mean of the scale estimate $E[\hat{\sigma}_n^2|\theta^*, H]$ should be derived first. Depending on complexity of the scale estimation procedure, such a derivation could be very complicated. An alternative approach to calculate the theoretical bias of the scale estimator is to use Monte Carlo simulation.

3 Calculation of Finite Sample Bias for Particular Robust Scale Estimators

3.1 Modified Selective Statistical Estimator (MSSE)

Bab-Hadiashar and Suter [6] have introduced a way of determining inliers that relies on finding the last *reliable* scale estimate as larger (sorted) residuals are added to the scale estimation. MSSE searches for the smallest sorted residual $r(k+1)$ that is T times larger than the least square scale estimate given by $r(1), r(2), \dots, r(k)$. By selecting $T = 2.5$, 98.76% of the inliers are expected to be detected in this scheme. In addition to T, there is another parameter in MSSE (denoted here by $k_{\min}$) which is the hypothesised minimum number of data samples in each single structure. MSSE scale estimate is formulated as below:

$$\hat{\sigma}^2_{\text{MSSE}} = 1.14 \sum_{i=1}^{k} r^2(i)/k \quad ; \quad k \geq k_{\min} \tag{2}$$

where k is the number of detected inliers and will be called the *cutting point* in this paper. The cutting point satisfies the following conditions:

$$\begin{array}{c} r^2(j+1) \leq T^2 \sum_{i=1}^{j} r^2(i)/j \qquad (k_{\min} \leq j \leq k-1) \\ r^2(k+1) > T^2 \sum_{i=1}^{k} r^2(i)/k. \end{array} \tag{3}$$

The constant factor 1.14 has been added to the original formulation of MSSE [6] to guarantee the consistency of the estimator when there is only one structure in the data samples.

The cutting point k depends on the residual values and the statistical mean of the scale estimate is given by:

$$E[\hat{\sigma}^2_{\text{MSSE}}] = \sum_{k=k_{\min}}^{n} p_k E[\hat{\sigma}^2_{\text{MSSE}} | k] \tag{4}$$

where $E[\hat{\sigma}^2_{\text{MSSE}}|k]$ is the conditional mean of the scale estimate for a known cutting point, and p_k is the probability of k being the cutting point and satisfying the conditions given in (3). Using equation (2), the conditional mean of the scale estimate can be derived as follows:

$$E[\hat{\sigma}^2_{\text{MSSE}}|k] = 1.14\, E\left[\sum_{i=1}^{k} r^2(i)\; |k\right]/k. \tag{5}$$

The expectation term $E\left[\sum_{i=1}^{k} r^2(i)\; |k\right]$ can be calculated by direct integration:

$$\begin{aligned} E\left[\sum_{i=1}^{k} r^2(i)\; |k\right] &= n\binom{n-1}{k-1} \int_{r_k=0}^{\infty} \int_{r_1=0}^{r_k} \cdots \int_{r_{k-1}=0}^{r_k} \int_{r_{k+1}=r_k}^{\infty} \int_{r_{k+2}=r_k}^{\infty} \cdots \int_{r_n=r_k}^{\infty} \\ &\quad \left(\sum_{i=1}^{k} r_i^2\right) \prod_{i=1}^{n} f^a(r_i)\, dr_n \cdots dr_{k+1} dr_{k-1} \cdots dr_1 dr_k \\ &= n\binom{n-1}{k-1} \int_{r_k=0}^{\infty} \left[1 - F^a(r_k)\right]^{n-k} \int_{r_1=0}^{r_k} \cdots \int_{r_{k-1}=0}^{r_k} \\ &\quad \left(\sum_{i=1}^{k} r_i^2\right) \prod_{i=1}^{k} f^a(r_i) dr_{k-1} \cdots dr_1 dr_k \\ &= n\binom{n-1}{k-1} \int_{r=0}^{\infty} \left[1 - F^a(r)\right]^{n-k} \left[F^a(r)\right]^{k-2} \\ &\quad \left\{ r^2 F^a(r) + (k-1)\left(\int_0^r \rho^2 f^a(\rho) d\rho\right)\right\} f^a(r) dr. \end{aligned} \tag{6}$$

Finding the cutting point among the sorted absolute residuals $\{r_{k_{\min}}, \ldots, r_n\}$ includes a search process to satisfy the conditions described in equation (3). Having n residuals, there are n choices to select a residual as r_k, $\binom{n-1}{k-1}$ choices to select the $k-1$ residuals that are less than it, $(k-1)!$ ways to order them in ascending order, and $n-k$ choices to select the residual that is immediately after r_k , called r_{k+1}. Thus, p_k is equal to $n(n-k)(k-1)!\binom{n-1}{k-1} = n!/(n-k-1)!$ times the integral of the joint pdf of residuals, over the subspace of residuals (D) in which the conditions described in equation (3) are satisfied. The probabilities p_k in equation (4) can be calculated by numerical computation of the following integral for each k:

$$p_k = \frac{n!}{(n-k-1)!} \int \cdots \int_D \prod_{i=1}^{n} f^a(r_i)\, dr_n \cdots dr_1. \tag{7}$$

Based on the conditions given in (3), the subspace D is the set of residual values within the following intervals:

$$\begin{array}{c}
r_j \in [r_{k+1}, +\infty] \quad ; \quad j > k+1 \\
r_{k+1} \in \left[\max\left(r_k, \sqrt{T^2 \sum_{i=1}^{k} r_i^2/k}\right), +\infty\right] \\
r_{j+1} \in \left[r_j, \sqrt{T^2 \sum_{i=1}^{j} r_i^2/j}\right] \quad ; \quad j = k_{\min}, \ldots, k-1 \\
r_{k_{\min}} \in \left[r_{k_{\min}-1}, \sqrt{T^2 \sum_{i=1}^{k_{\min}-1} r_i^2/(k_{\min} - T^2)}\right] \\
\vdots \\
r_{k_{\min}-l+1} \in \left[r_{k_{\min}-l}, T^2 \sum_{i=1}^{k_{\min}-l} r_i^2/(k_{\min} - lT^2)\right] \\
r_{k_{\min}-l} \in [r_{k_{\min}-l-1}, +\infty] \\
\vdots \\
r_2 \in [r_1, +\infty] \\
r_1 \in [0, +\infty]
\end{array} \tag{8}$$

where $l = \lceil \frac{k_{\min}}{T^2} \rceil - 1$. Knowing the number of samples n, $k_{\min}$ and the pdf of absolute residuals $f^a(r)$, the bias of MSSE can be calculated using equations (6) and (7). The residual density function $f^a(r)$ depends on the distribution of the different structures and outliers in any given problem. We will elaborate more on formulating the residual density in section 4.

3.2 Residual Consensus (RESC) Estimator

In RESC technique, the scale of the inliers is estimated by directly calculating [4]:

$$\hat{\sigma}^2_{\text{RESC}} = \xi \sum_{i=1}^{\nu} (ih_i^c \delta - \overline{h}^c)^2 / (\sum_{i=1}^{\nu} h_i^c - 1) \tag{9}$$

where $\overline{h}^c$ is the mean of all residuals included in the compressed histogram, δ is the bin size of the compressed histogram, ξ is a correction factor for the approximation introduced by rounding the residuals in a bin of histogram to $i\delta$, and ν is the number of bins. Wang and Suter [5] have reported that this method overestimates the scale and have proposed the following alternative formulae:

$$\hat{\sigma}^2_{\text{RESC}} = \sum_{i=1}^{n_c} \left(r_i - \overline{h}^c\right)^2 / (\sum_{i=1}^{\nu} h_i^c - 1) \tag{10}$$

where n_c is the number of data points in the compressed histogram (detected inliers). In [4], the residual of each inlier is assumed to belong to one of the first ν bins in the histogram, $\{h_i^c; i = 1, \cdots, \nu\}$, for which the ratio h_i^c/h_1^c is greater than or equal to $\exp(-\gamma^2/2)$. The bin size of the histogram is automatically determined by the ordered statistics of the residuals: $\delta = r_{(\lceil \rho n \rceil)}$ where ρ is a fixed parameter. The inputs of the algorithm are the residuals and the parameter ρ (0.12 in [4]).

Because of the complication involved in mathematical formulation of the statistical distribution of the detected residuals and the statistical mean of the scale estimate, direct calculation of the bias of RESC (unlike MSSE) is not straightforward. Instead, we use Monte Carlo simulation as explained in section 4.

3.3 Two-Step Scale Estimator (TSSE)

Wang and Suter [5] have devised a technique to estimate the scale of noise based on mean shift method and its modified form called "mean shift valley algorithm". In the first step, TSSE uses mean shift (with initial centre zero) to find a local peak close to zero on the residual density curve, then uses the mean shift valley algorithm to find the valley (local minimum) next to that peak. The residuals of structures other than the target structure are disregarded as they lie outside the obtained valley [5]. In the second step, the scale of noise is estimated by a median scale estimator using the points within the band centred at the local peak extending to the valley. Similar to RESC, the statistical mean of the scale estimate given by TSSE is too complicated to be mathematically formulated. In this paper, finite sample bias of TSSE is evaluated by Monte Carlo simulations.

4 Numerical Simulation

In our numerical analysis, we consider various data scenarios each including two parallel hyper-planes in p-dimensional space (forming one step in two and three-dimensional cases). For MSSE, the theoretical bias of the estimator can be numerically calculated by using equations (1) and (4)-(7). If σ is the true scale of noise and the height of the p-dimensional step (the algebraic distance between the two hyper-plane) is $\mu\sigma$, then the pdf of absolute residuals (corresponding with the hypothesised fit θ^*) is given by:

$$f^a(r) = \frac{2}{\sqrt{2\pi}\sigma}\left[\epsilon \exp(-\frac{r^2}{2\sigma^2}) + (1-\epsilon)\exp(-\frac{(r-\mu\sigma)^2}{2\sigma^2})\right] \quad ; \quad r \geq 0 \qquad (11)$$

where ϵ is the ratio of inlier samples (target structure) and the inlier noise is assumed to be normally distributed. It is important to note that the above distribution does not depend on the dimension of data samples p because as long as the two structures are parallel, the residual distribution comprises two normal populations linearly combined with each other.

The bias of estimator, as defined in (1), is scale-invariant and the only parameter in equation (11) to be set for calculating the bias is the relative step height μ. In our simulations, we examine different cases with $\mu = 3, \cdots$, or 10. Each case is examined with 30% inliers included in data ($\epsilon = 0.3$). In our previous work, we have shown that

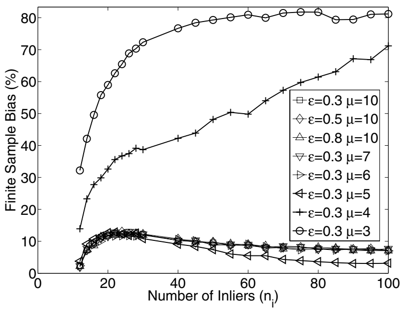


Fig. 3. Theoretical finite sample bias of MSSE

MSSE, RESC and TSSE are asymptotically unbiased (consistent) for μ =10, both in theory and practice [8]. To focus on the effect of inlier ratio on finite sample bias, we examine the other cases ϵ =0.5, 0.8 both with μ =10 in this paper. A minimum number of $k_{\min} = 10$ inlier samples are assumed to exist in each structure in our simulation studies.

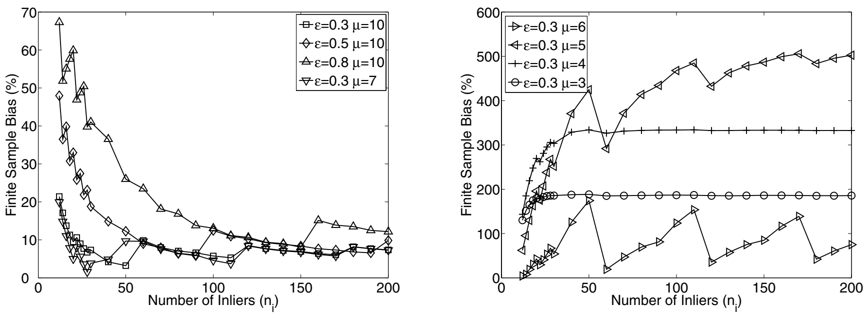
Fig. 3 shows the results for the bias of MSSE applied in each case with different number of data samples. As a robust estimator, MSSE is expected to detect outliers and only incorporates the inliers in its scale estimation process. Therefore, the finite sample bias of the estimator depends on the number of inliers (not on the total number of data samples) and in Fig. 3 the bias measures are plotted versus the number of inliers $n_i = \lceil \epsilon n \rceil$.

To study the finite sample bias of RESC and TSSE in similar data scenarios, we have utilised Monte Carlo method. For each data scenario (with the same μ and ϵ parameters as chosen for MSSE) we generate 10^6 vectors of residuals, each in the following form:

$$\mathbf{r} = [\mathbf{r}_1^T \ \mathbf{r}_2^T]^T \ ; \ \mathbf{r}_1 \sim \mathcal{N}(0, \sigma^2 I_{n_i}) \ ; \ \mathbf{r}_2 \sim \mathcal{N}(\mu\sigma, \sigma^2 I_{n_o}) \tag{12}$$

where $n_o = n - n_i$ and for each vector, $\mathbf{r}_1$ and $\mathbf{r}_2$ are generated by a random number generator. Thus, 10^6 numerical instances of the residuals are generated with their absolute values distributed according to equation (11). Each vector of residuals is given as an input to the RESC and TSSE scale estimation schemes (described in subsections 3.2 and 3.3) and a scale estimate is calculated by each estimator. The average of all 10^6 scale estimates is a close approximation to the statistical mean of the scale estimate $E[\hat{\sigma}_n^2|\theta^*, H]$ and by substituting it in equation (1), the finite sample bias of the RESC and TSSE are numerically calculated in the given scenario. Figures 4 and 5 show the results of Monte Carlo simulation for the bias of RESC and TSSE, respectively, in the same data scenarios studied for MSSE. Figures 3, 4 and 5 show a number of important differences in the performance of MSSE, RESC and TSSE when a limited number of data samples belonging to the target structure are available. These differences are mainly related to how the estimator detects the outliers and extracts the scale of noise from the inliers.

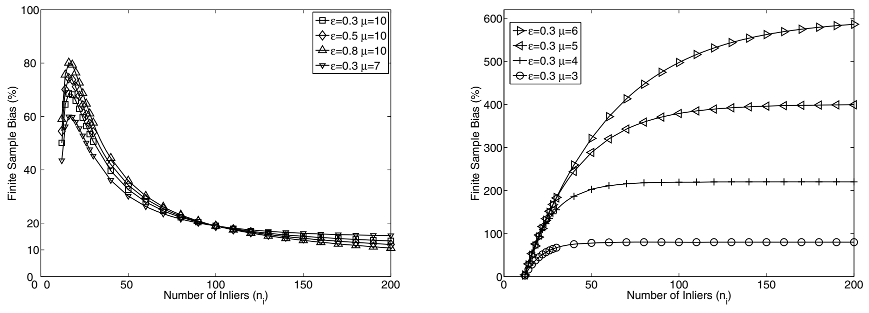
Fig. 3 shows that MSSE is heavily biased when the relative distance of the two parallel structures, μ, is less than five. For $\mu > 5$, MSSE is asymptotically unbiased



(a) (b)

Fig. 4. Bias of RESC with: (a) close and (b) Very close parallel hyper-planes

and its finite sample bias is always less than 20%, and if $n > 45$ its bias is less than 10%. Surprisingly, with $\mu = 5$ the bias is smaller than the cases with larger distances between the structures. The main reason is that the cutting point k in equation (2) is usually lower than the true number of inliers and the MSSE scale is underestimated. But, when two structures with an algebraic distance of 5σ are considered, some of the data samples belonging to the second structure are mixed with the data from the target structure (note that the cutting threshold T in (3) is 2.5) and the cutting point is closer to the true number of inliers. Therefore, the estimator is less biased compared to other cases with larger μ values.



(a) (b)

Fig. 5. Bias of TSSE with: (a) close and (b) very close parallel hyper-planes

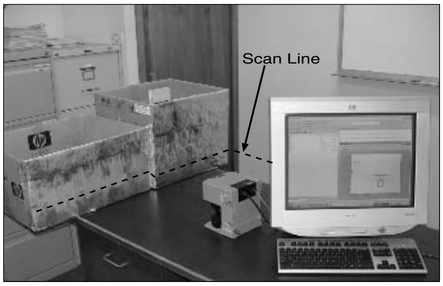
Figures 4 and 5 show that RESC and TSSE are asymptotically unbiased for $\mu > 6$ and their finite sample biases are larger than MSSE. More precisely, while the bias of MSSE does not exceed 20%, the bias of RESC and TSSE can be as large as 68% and 83% respectively. Indeed, at least 72 and 105 inlier samples are required for the bias of RESC and TSSE to be less than 20%, respectively, and more than 200 inlier samples for their bias to be less than 10%. The main reason for the poor performance of RESC and

TSSE in terms of their finite sample bias (compared to MSSE) is that they detect the outliers based on an estimate of the density of residuals (histogram technique in RESC and kernel density estimation in TSSE), while MSSE directly uses the least square scale estimate of the hypothesised inliers to find the cutting point. Therefore, due to the finite sample errors involved in density estimation, RESC and TSSE are more likely to underestimate or overestimate the cutting point and the resulting scale estimate.

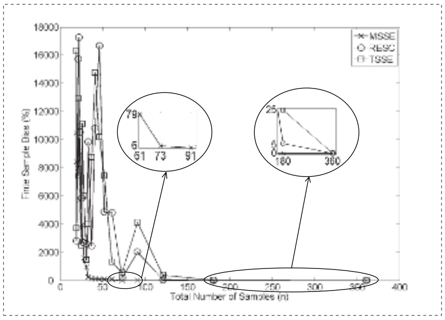
5 Experimental Results

To study the finite sample bias of robust estimators in practice, an indoor range segmentation experiment was conducted using a SICK laser measurement system (LMS200-30106). Fig. 6(a) shows a picture of the experimental setup where two boxes in front of the laser scanner constitute two parallel surfaces to be segmented in the experiment. One of the two boxes was considered the target structure, true scale of measurement noise was 4.3mm and the distance between the parallel surfaces of the two boxes was around 5cm (corresponding with $\mu = 11.6$). Our previous study [8] has shown that MSSE, RESC and TSSE are asymptotically unbiased for this ratio of distance to the scale of noise. Setting the measurement range to 0-180° with 0.5° resolution, each scan provided 361 data samples which is large enough for the purpose of this study.

Several data sets were built from the 361 range data samples. The first data set contains all samples and corresponds with a resolution of 0.5° and $n = 361$. The second data set contains one every two of the samples, corresponding with a resolution of 1.0° and $n = 181$. The third data set contains one every three of the samples, corresponding with a resolution of 1.5° and $n = 121$. Similarly, smaller data sets were realised for bias analysis. For each scale estimator, the empirical bias was calculated using equation (1) with $E[\hat{\sigma}_n^2|\theta^*, H]$ replaced with the estimated squared scale $\hat{\sigma}^2$ and $\sigma = 4.3$mm. The results are plotted versus the number of data samples in Fig. 6(b). MSSE appears to perform better than RESC and TSSE, as its bias is no more than 6% for $n \geq 73$, while for RESC at least 180 samples are required to have a bias of 5% or less, and the bias of TSSE is under 25% only when all 360 data samples are applied. These results



(a)



(b)

Fig. 6. (a) Experimental setup (b) Finite sample bias of MSSE, RESC and TSSE are in line with the theoretical results presented in section 4. The slight difference is mainly because the actual distribution of the inlier noise is not exactly normal as assumed in our analysis and simulations. In addition, the single scan contains only one instance of the underlying population of possible measurements. The analytical results, on the other hand, are based on the statistical average of the noise scale over all possible measurements.

6 Conclusions

We have introduced a scale invariant performance measure that signifies the finite sample bias of an estimator. We have also either formulated or numerically evaluated (using Monte Carlo simulations) this measure for three state-of-the-art high breakdown robust estimators: RESC, MSSE and TSSE. As the estimators are mainly developed to assist the segmentation task, the main emphasis has been on evaluating the finite sample bias of those estimators for data sets containing at least two near but distinct structures for various inlier ratios and different normalised distances between the two structures. Our results show that MSSE outperforms RESC and TSSE in terms of its small sample bias as it directly processes the residuals and detects the outliers. The other two estimators use the distribution of the data which can't be reliably recovered for small samples and their finite sample bias performances appear to be similar.

References

1. Bab-Hadiashar, A., Suter, D.: Robust Optic flow computation. Int. J. Computer Vision. Vol. 29 No. 1 (1998) 59-77.
2. Meer, P.: Robust techniques for computer vision. In Medioni, G., Kang, S.B., eds.: Emerging Topics in Computer Vision, Prentice Hall (2004) 107-190
3. Torr, P.H.S.: Geometric motion segmentation and model selection. Philosophical Trans. of the Royal Society A (1998) 1321-1340
4. Yu, X., Bui, T.D., Krzyzak, A.: Robust estimation for range image segmentation and reconstruction. IEEE Trans. PAMI. Vol. 16 No.5 (1994) 530-538
5. Wang, H., Suter, D.: Robust adaptive-scale parametric model estimation for computer vision. IEEE Trans. PAMI. Vol. 26 No. 11 (2004) 1459-1474
6. Bab-Hadiashar, A., Suter, D.: Robust segmentation of visual data using ranked unbiased scale estimator. ROBOTICA. Vol. 17 (1999) 649-660
7. Stewart, C.V.: Bias in robust estimation caused by discontinuities and multiple structures. IEEE Trans. PAMI. Vol. 19 No. 8 (1997) 818-833
8. Authors: Consistency analysis of robust scale estimators for multi-structural visual data. Submitted to IEEE Trans. PAMI.

Flexible Segmentation and Smoothing of DT-MRI Fields Through a Customizable Structure Tensor

Thomas Schultz[1], Bernhard Burgeth[2], and Joachim Weickert[2]

[1] MPI Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany
schultz@mpi-inf.mpg.de
[2] Mathematical Image Analysis Group, Faculty of Mathematics and Computer Science, Saarland University, 66041 Saarbrücken, Germany
{burgeth, weickert}@mia.uni-saarland.de

Abstract. We present a novel structure tensor for matrix-valued images. It allows for user defined parameters that add flexibility to a number of image processing algorithms for the segmentation and smoothing of tensor fields. We provide a thorough theoretical derivation of the new structure tensor, including a proof of the equivalence of its unweighted version to the existing structure tensor from the literature. Finally, we demonstrate its advantages for segmentation and smoothing, both on synthetic tensor fields and on real DT-MRI data.

1 Introduction

In recent years, second-order tensor fields have received increasing attention. This is partly due to the now widely-used diffusion tensor magnetic resonance imaging (DT-MRI) modality that uses a diffusion tensor in each voxel to describe the self-diffusion of water molecules [1]. The methods presented in this paper have been developed with an eye on the processing of DT-MRI data, but can be used wherever real-valued, symmetric 3×3 matrix fields arise.

To approach the smoothing and segmentation of such fields, Feddern et al. [2,3] have proposed an extension of some well-known curvature-based partial differential equations (PDEs), like mean curvature motion and active contour models, to the tensor case. Other methods for tensor image regularization [4,5,6,7] and segmentation [8,9,10,11] have been suggested. However, all of them use a fixed distance measure on the tensors. Therefore, none of the existing approaches allow the user to emphasize the relevance of particular properties of the diffusion tensor (i.e., overall diffusivity, anisotropy, and orientation) for a given application.

Feddern et al. derive a structure tensor for tensor-valued images. Subsequently, they define generalized level lines as integral lines of its minor eigenvector field and generalized gradient magnitude as the structure tensor trace.

Our present work uses a decomposition of the tensor field gradient which has been suggested by Kindlmann [12] to replace this structure tensor with a new formulation that has user defined parameters. When they are all set to one, the new formulation is equivalent to the previous one. However, we will show that the option to weight its individual terms is crucial for some specific applications.

The paper is organized as follows: Section 2 revises the previous work by Kindlmann which serves as the basis of our re-formulation, presented in Section 3. A proof of the equivalence to the previous structure tensor is given in Section 4. In Section 5, example applications on synthetic and real DT-MRI data follow, before Section 6 concludes the paper.

2 Projected Tensor Gradients

Let Sym_3 denote the six-dimensional vector space of symmetric, real-valued 3×3 matrices and let $\mathbf{D}$ be a field of Sym_3 matrices over $\mathbb{R}^3$. Then, the gradient $\nabla\mathbf{D}$ of the field is a $3 \times 3 \times 3$ third-order tensor which we will index such that $(\nabla\mathbf{D})_{ijk} = \frac{\partial D_{jk}}{\partial x_i}$. Thus, $\nabla\mathbf{D}$ can be thought of as a three-vector of second-order tensors, expressing the partial derivatives of $\mathbf{D}$ in each image direction.

Kindlmann's contribution in [12] is to decompose $\nabla\mathbf{D}$ into parts that correspond to changes in three tensor invariants which cover changes in shape, as well as the parts of $\nabla\mathbf{D}$ that correspond to rotations around each eigenvector.

The invariants he uses to describe tensor shape are derived from the moments of the eigenvalues λ_1, λ_2, and λ_3. In the context of DT-MRI, $\mu_1 := \frac{1}{3}\sum_i \lambda_i$, the eigenvalue mean, is a measure of *bulk diffusivity.* The eigenvalue variance $\mu_2 := \frac{1}{3}\sum_i(\lambda_i - \mu_1)^2$ measures *diffusion anisotropy.* The eigenvalue skewness $\alpha_3 := \mu_3/\sqrt{\mu_2^3}$ (with $\mu_3 := \frac{1}{3}\sum_i(\lambda_i - \mu_1)^3$) reflects the *type of anisotropy.* It is a dimensionless quantity with range $\left[-1/\sqrt{2}, 1/\sqrt{2}\right]$, where $\alpha_3 = -1/\sqrt{2}$ indicates a perfectly planar tensor and $\alpha_3 = 1/\sqrt{2}$ a perfectly linear one.

If we consider these invariants as scalar-valued functions over Sym_3, their gradient is a map from Sym_3 to Sym_3. We will denote this gradient of an invariant J as the *invariant gradient* $\boldsymbol{\nabla} J$, marked by a boldface $\boldsymbol{\nabla}$.

Kindlmann decomposes the tensor field gradient $\nabla\mathbf{D}$ by projecting it onto the invariant gradients. In order to avoid undesired scaling in this step, he first normalizes $\boldsymbol{\nabla} J$ with respect to the tensor scalar product $\mathbf{A} : \mathbf{B} := \sum_{i,j} a_{ij}b_{ij}$. Let $\|\mathbf{D}\| := \sqrt{\mathbf{D} : \mathbf{D}}$ denote the associated Frobenius norm. If $\|\boldsymbol{\nabla} J\| > 0$, its normalized version is $\hat{\boldsymbol{\nabla}} J := \boldsymbol{\nabla} J / \|\boldsymbol{\nabla} J\|$. However, $\boldsymbol{\nabla}\mu_2$ vanishes when $\mu_2 = 0$, and $\boldsymbol{\nabla}\alpha_3$ vanishes when α_3 reaches one of its extrema as well as in the isotropic case ($\mu_2 = 0$), for which α_3 is undefined.

Kindlmann derives expressions for the invariant gradients[1] and suggests auxillary constructs to ensure that $\hat{\boldsymbol{\nabla}}\mu_1$, $\hat{\boldsymbol{\nabla}}\mu_2$ and $\hat{\boldsymbol{\nabla}}\alpha_3$ are always orthonormal and span the space of changes in tensor shape, even if some of the underlying invariant gradients are undefined. As a consequence, the directions of $\hat{\boldsymbol{\nabla}}\mu_2$ and $\hat{\boldsymbol{\nabla}}\alpha_3$ are arbitrary and exchangeable when $\mu_2 = 0$.

[1] The exact formulas are not required to understand this paper. The interested reader can find them in [12, p. 73].

The *projected gradient* ∇J is obtained by taking the tensor scalar product of $\hat{\boldsymbol{\nabla}} J$ with each of the three Sym_3 tensors within $\nabla\mathbf{D}$:

$$(\nabla J)_i := \sum_{j,k} (\hat{\boldsymbol{\nabla}} J)_{jk} (\nabla \mathbf{D})_{ijk} \tag{1}$$

∇J is a vector in $\mathbb{R}^3$, expressing for each of the three spatial directions the amount of tensor change that corresponds to changes in the invariant J.

To cover changes in orientation, Kindlmann calculates *rotation tangents* ($\boldsymbol{\Phi}_1, \boldsymbol{\Phi}_2, \boldsymbol{\Phi}_3 \in Sym_3$) as the changes in $\mathbf{D}$ caused by infinitesimal rotations around its eigenvectors. Subsequently, the effect of finite rotations is approximated by adding some scalar multiple of the normalized tangents $\hat{\boldsymbol{\Phi}}_i$ [12, p. 87].

Kindlmann shows that the rotation tangents are orthogonal both to each other and to the invariant gradients defined above. The rotation tangents are used in the same manner as before to define projected tensor field gradients $\nabla\phi_i$.

3 The New Structure Tensor

The structure tensor used by Feddern et al. [3] is defined as follows:

$$\mathbf{J}_{\text{orig}}(\nabla\mathbf{D}_\sigma) := \sum_{i=1}^{3}\sum_{j=1}^{3} \nabla(\mathbf{D}_\sigma)_{i,j}\ \nabla^T(\mathbf{D}_\sigma)_{i,j} \tag{2}$$

The indexing with σ indicates that the gradient is calculated from a Gaussian-smoothed version of the original tensor field, to allow noise-scale pre-processing. This definition can be considered an extension of Di Zenzo's approach for vector-valued images [13]. Its advantages are that it makes use of the full tensor information, it is rotationally invariant, and it has proven to work well in practice.

Equation (2) is based on the gradients of the nine tensor channels. Our alternative approach uses the six projected gradients from Section 2 instead. As these are physically meaningful, their individual influence can reasonably be weighted via user coefficients w_*[2].

However, we now need to handle cases in which the invariant gradients become ill-defined. We will tackle this problem with functions ψ_* that calculate effective weights from the user-controlled parameters w_*.

For isotropic tensors ($\mu_2 = 0$), $\hat{\boldsymbol{\nabla}}\mu_2$ and $\hat{\boldsymbol{\nabla}}\alpha_3$ are arbitrary and exchangeable. We reason that in the case of perfect isotropy, it does not make sense to speak of changes in the *type* of anisotropy, so all tensor change that gets projected on the span of $\hat{\boldsymbol{\nabla}}\mu_2$ and $\hat{\boldsymbol{\nabla}}\alpha_3$ should be attributed to changes in variance.

To make a smooth transition towards this case, we use the fractional anisotropy (FA), a common anisotropy measure in the context of DT-MRI [14]:

$$\text{FA} := \sqrt{\frac{3}{2}}\frac{\|\mathbf{D} - \mu_1\mathbf{I}\|}{\|\mathbf{D}\|} = \frac{3}{\sqrt{2}}\frac{\sqrt{\mu_2}}{\|\mathbf{D}\|} \tag{3}$$

[2] With the asterisk $*$, we refer to all possible indices of a variable.

where $\mathbf{I}$ is the unit matrix. It is clear from Equation (3) that FA $= 0$ if $\mu_2 = 0$, so we define ψ_{α_3} such that it tends to w_{μ_2} as FA $\rightarrow 0$:

$$\psi_{\alpha_3}(\mathrm{FA}; w_{\mu_2}, w_{\alpha_3}) := \begin{cases} w_{\alpha_3} & \text{if FA} > \epsilon \\ \left(\frac{\mathrm{FA}}{\epsilon} - 1\right)^2 w_{\mu_2} + \left[1 - \left(\frac{\mathrm{FA}}{\epsilon} - 1\right)^2\right] w_{\alpha_3} & \text{else} \end{cases}$$

Making a smooth transition requires to introduce a parameter ϵ that defines the threshold starting from which we fully rely on the gradient direction $\hat{\boldsymbol{\nabla}}\alpha_3$. In our experiments, a value of $\epsilon = 0.1$ worked well.

Further singularities occur when dealing with changes in orientation. Since the two ill-defined eigenvectors of a perfectly linear ($\alpha_3 = 1/\sqrt{2}$) or planar ($\alpha_3 = -1/\sqrt{2}$) tensor are exchangeable, the effect of rotating the tensor around any of them may just as well be attributed to the other one. Thus, the ψ_{ϕ_i} are designed to let them share the total amount of rotation around any of them by averaging their weights in the limit case:

$$\psi_{\phi_1}(\mathrm{FA}, \alpha_3; w_{\phi_1}, w_{\phi_2}) = \begin{cases} w_{\phi_1} & \text{if FA} = 0 \text{ or } \alpha_3 > 0 \\ \left(1 - \alpha_3^2\right) w_{\phi_1} + \alpha_3^2 w_{\phi_2} & \text{else} \end{cases}$$

$$\psi_{\phi_2}(\mathrm{FA}, \alpha_3; w_{\phi_1}, w_{\phi_2}, w_{\phi_3}) = \begin{cases} w_{\phi_2} & \text{if FA} = 0 \\ \left(1 - \alpha_3^2\right) w_{\phi_2} + \alpha_3^2 w_{\phi_1} & \text{if FA} > 0 \text{ and } \alpha_3 < 0 \\ \left(1 - \alpha_3^2\right) w_{\phi_2} + \alpha_3^2 w_{\phi_3} & \text{else} \end{cases}$$

$$\psi_{\phi_3}(\mathrm{FA}, \alpha_3; w_{\phi_2}, w_{\phi_3}) = \begin{cases} w_{\phi_3} & \text{if FA} = 0 \text{ or } \alpha_3 < 0 \\ \left(1 - \alpha_3^2\right) w_{\phi_3} + \alpha_3^2 w_{\phi_2} & \text{else} \end{cases}$$

With these definitions, the new structure tensor reads:

$$\begin{aligned} \mathbf{J}_{\mathrm{new}}(\nabla\mu_1, \nabla\mu_2, \nabla\alpha_3, \nabla\phi_1, \nabla\phi_2, \nabla\phi_3; \mathrm{FA}, \alpha_3) :=& \\ & w_{\mu_1} \nabla\mu_{1,\sigma} \nabla\mu_{1,\sigma}^T + w_{\mu_2} \nabla\mu_{2,\sigma} \nabla\mu_{2,\sigma}^T + \\ & \psi_{\alpha_3}(\mathrm{FA}; w_{\mu_2}, w_{\alpha_3}) \nabla\alpha_{3,\sigma} \nabla\alpha_{3,\sigma}^T + \\ & \sum_{i=1}^{3} \psi_{\phi_i}(\mathrm{FA}, \alpha_3; w_{\phi_1}, w_{\phi_2}, w_{\phi_3}) \nabla\phi_{i,\sigma} \nabla\phi_{i,\sigma}^T \end{aligned} \tag{4}$$

It is rotationally invariant for arbitrary sets of weights and uses the full tensor information when all weights are non-zero.

4 Equivalence to the Previous Structure Tensor

In its unweighted form ($w_* = 1$), the new structure tensor $\mathbf{J}_{\mathrm{new}}$ is equivalent to the one used by Feddern et al., $\mathbf{J}_{\mathrm{orig}}$. This fact ensures that our new structure tensor has all the desirable properties of the established one.

Proof. We can write element (i, j) of the original structure tensor $\mathbf{J}_{\text{orig}}$ (2) in terms of the tensor scalar product:

$$(\mathbf{J}_{\text{orig}})_{ij} = \nabla \mathbf{D}_i : \nabla \mathbf{D}_j \tag{5}$$

Note that the normalized invariant gradients and rotation tangents defined by Kindlmann form an orthonormal basis of Sym_3. If we denote the kth of these basis vectors by $\mathbf{B}^k$, we can write element (i, j) of the new structure tensor $\mathbf{J}_{\text{new}}$ (4) correspondingly as

$$(\mathbf{J}_{\text{new}})_{ij} = \sum_k \left(\nabla \mathbf{D}_i : \mathbf{B}^k\right) \left(\nabla \mathbf{D}_j : \mathbf{B}^k\right) \tag{6}$$

For the sake of simplicity, we are going to embed the tensors isometrically in $\mathbb{R}^6$. This can be done by using the six non-redundant tensor channels as components of the vector, where non-diagonal elements are multiplied by $\sqrt{2}$.

Let $\nabla \mathbf{d}_i$ and $\mathbf{b}^k$ be the embedded versions of $\nabla \mathbf{D}_i$ and $\mathbf{B}^k$, respectively. Then, we can re-write Equation (6) and reorder the terms as follows:

$$\begin{aligned}(\mathbf{J}_{\text{new}})_{ij} &= \sum_k \left(\nabla \mathbf{d}_i \cdot \mathbf{b}^k\right) \left(\nabla \mathbf{d}_j \cdot \mathbf{b}^k\right) \\ &= \sum_l \left(\sum_k (b_l^k)^2\right) \nabla d_{il} \nabla d_{jl} + \sum_{\substack{l,m \\ l \neq m}} \left(\sum_k b_l^k b_m^k\right) \nabla d_{il} \nabla d_{jm}\end{aligned} \tag{7}$$

If we arrange the $\mathbf{b}^k$ as rows of a matrix, the resulting matrix is orthogonal, because the $\mathbf{B}^k$ were orthonormal and our mapping preserved the scalar product. Thus, the column vectors of the matrix will also be orthonormal: $\sum_k b_l^k b_m^k = \delta_{lm}$.

With this result, Equation (7) reduces to $\nabla \mathbf{d}_i \cdot \nabla \mathbf{d}_j$, which is by our definition equivalent to Equation (5). □

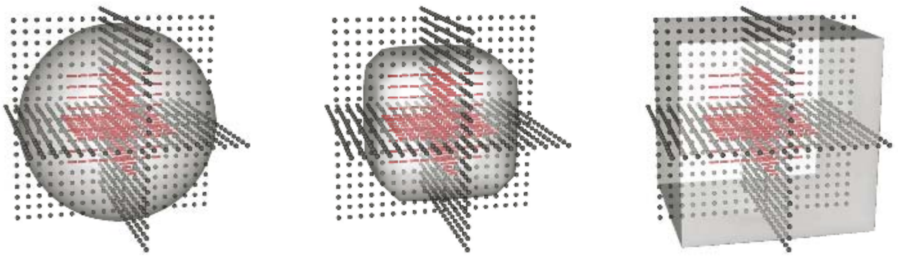
5 Applications

5.1 Application to Segmentation

We will now demonstrate the advantages of the weightable structure tensor in the context of a geodesic active contour model [15,16] for interactive segmentation. This model allows the user to provide the approximate position and shape of the object that is to be segmented. Consequently, the contour moves to the exact boundary of the object, based on edge information from the image.

For tensor-valued images $\mathbf{f}$, Feddern et al. [3] suggest to use the structure tensor trace as an edge detector. A simple implementation of the segmentation model is then given by embedding the initial contour as a zero level set into a function u_0 (via a distance transformation) and evolving it under the PDE

$$\partial_t u = |\nabla u| \operatorname{div} \left(g(\operatorname{tr} \mathbf{J}(\nabla \mathbf{f}_\sigma)) \frac{\nabla u}{|\nabla u|} \right) \tag{8}$$


(a) Initialization of the active surface, between two edges

(b) Without weighting, the inner box gets segmented

(c) Setting $w_{\mu_1} := 10$ leads to a segmentation of the outer box

Fig. 1. Active surface segmentation of a synthetic data set ($t = 500$). Varying the weights draws the segmentation towards specific types of edges in the data.

which simultaneously regularizes the surface and draws it towards edges in the image $\mathbf{f}$. g is a non-increasing stopping function. In our experiments, we used the following diffusivity [17, p. 114]:

$$g(s^2) := \begin{cases} 1 & \text{if } s^2 = 0 \\ 1 - \exp\left(\frac{-3.31488}{(s^2/\lambda^2)^4}\right) & \text{if } s^2 > 0 \end{cases} \tag{9}$$

The evolution of Equation (8) is stopped at a time t, when u no longer changes significantly. The segmentation result is extracted as the zero level set of u. Unlike the version presented in [3], our implementation of (8) works in 3D, so we will refer to it as an *active surface model.*

We first applied this method to the synthetic dataset shown in Figure 1. It consists of two nested boxes of different materials: The tensors in the outer box are isotropic, but have the same trace as the linear ones in the inner box. Thus, two types of edges arise: A change of tensor trace between air and the outer box and a change of anisotropy between the outer and the inner box.

Figure 1 shows the setup using superquadric glyphs [18] on three orthogonal slices. When we initialize the active surface to a sphere that lies between the two boundaries (Figure 1(a)), the unweighted structure tensor leads to a segmentation of the inner box (Figure 1(b)). By increasing the weight of changes in trace ($w_{\mu_1} := 10$), we can guide the segmentation to the outer box.

We also segmented the corpus callosum (CC), a major white-matter structure, in a real DT-MRI dataset. Figure 2(a) shows the initialization of the active surface to an ellipsoid, superimposed on a sagittal slice of the data in the standard xyz-RGB eigenvector color coding. It encompasses the structure of interest, which appears as red in the color image, or as dark in halftones.

The original, unweighted structure tensor basically leads to a segmentation of the ventricle (Figure 2(b), the ventricle is shown in white). The CC is distinguished from its neighborhood by its anisotropy ($w_{\mu_2} := 1$) and major

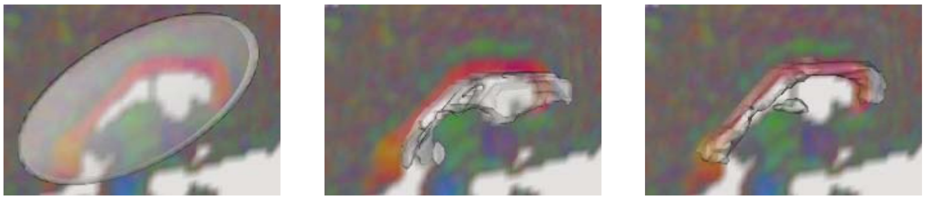

(a) Initialization of the active surface (b) The unweighted structure tensor segments the ventricle (c) The weights allow to capture the corpus callosum

Fig. 2. Active surface segmentation of a DT-MRI data set ($t = 1200$). Weighting the new structure tensor allows to specify the structure of interest.

eigenvector orientation ($w_{\phi_2} := w_{\phi_3} := 0.2$). Setting all other weights zero allows for a plausible segmentation of this structure, shown in Figure 2(c).

A small structure inferior to the CC also got segmented. Although it differs from the CC by its global orientation, we cannot differentiate it in our model, which only considers local variations in tensor value. In practice, a simple connected component analysis would be sufficient to remove the undesired island.

For the numerical implementation, we have used the additive operator splitting (AOS) scheme as described by Weickert and Kühne [19]. It allowed us to solve Equation (8) on the real DT-MRI dataset (grid size $74 \times 95 \times 80$) for $t = 1200$ in 40 seconds on a 2 GHz dual-core Athlon 64 processor; the synthetic examples took two seconds each.

5.2 Application to Smoothing

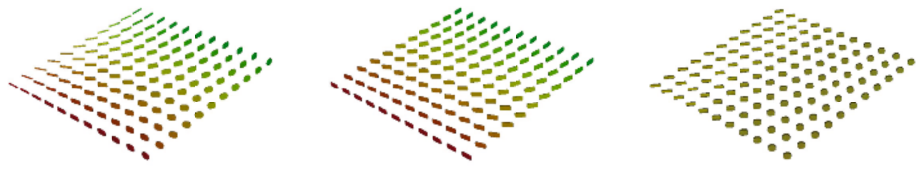
Feddern et al. [3] also generalize the *self-snakes* smoothing process initially proposed by Sapiro [20] to the tensor-valued case. In 3D, it can be written as a system of six coupled PDEs for the individual tensor channels,

$$\partial_t u_{i,j} = g(\operatorname{tr}\mathbf{J}(\nabla\mathbf{u}_\sigma))\,(\partial_{\mathbf{vv}}u_{i,j} + \partial_{\mathbf{ww}}u_{i,j}) + \nabla^T(g(\operatorname{tr}\mathbf{J}(\nabla\mathbf{u}_\sigma)))\nabla u_{i,j} \qquad (10)$$

with $u_{i,j}(x, y, z, 0) = f_{i,j}(x, y, z)$ as the initial condition. The vectors $\mathbf{v}$ and $\mathbf{w}$ denote the minor and medium eigenvectors of the structure tensor $\mathbf{J}$, respectively. Equation (10) leads to a smoothing along the plane spanned by $\mathbf{v}$ and $\mathbf{w}$; this is analogous to the smoothing along level surfaces in scalar-valued mean curvature motion (MCM). The diffusivity function g stops the smoothing process at important image features.

Our new structure tensor allows to steer this process. We first demonstrate this with the synthetic dataset from Figure 3(a), in which the tensors vary continuously in shape (from linear to planar) and orientation (rotation by 90°). In this example, we set $g := 1$ to obtain purely MCM-like smoothing.

Setting the weights of $\mathbf{J}_{\text{new}}$ allows to specify which features we would like to preserve. If we set $w_{\phi_*} := 1$ (all other weights zero), the result in Figure 3(b) shows that the shapes are completely averaged at $t = 250$, while the orientation



(a) A tensor field that varies in shape and orientation

(b) Result of orientation-preserving smoothing

(c) Result of shape-preserving smoothing

Fig. 3. Mean Curvature Motion-like smoothing of a synthetic data set ($t = 250$). Customizing the structure tensor allows to select which features should be preserved.

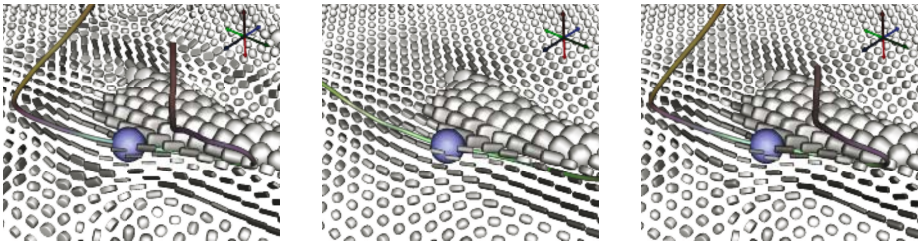
has been preserved. On the other hand, we can set only $w_{\mu_2} := w_{\alpha_3} := 1$ to preserve shape and average the orientation, as shown in Figure 3(c).

In our experiments on real DT-MRI data, we found that for many smoothing tasks, $\mathbf{J}_{\mathrm{orig}}$ already works well in its default configuration. Figure 4 presents a situation in which its behaviour can still be improved by changing the weights.

Figure 4(a) shows two touching fiber tracts. One of them lies in the depicted sagittal plane, the other one (shown by an integral curve of major eigenvectors, started at the position marked by the ball) touches the plane almost tangentially. It is a delicate task to smooth the dataset without mixing these two tracts.

Figure 4(b) shows that after applying self-snakes with $\mathbf{J}_{\mathrm{orig}}$, the tangentially touching tract has been lost at $t = 5$. To preserve the directions of both tracts, we configured $\mathbf{J}_{\mathrm{new}}$ to concentrate on orientation ($w_{\phi_*} := 1$) and to prevent influence of the nearby ventricle ($w_{\mu_1} := 0.1$, all other weights zero). Figure 4(c) shows that this configuration makes it possible to preserve both tracts.

In this experiment, we set g to the diffusivity function (9). As the trace of different structure tensors does not in general lie on the same order of magnitude, we select the contrast parameter λ^2 as 0.01 times the maximum structure tensor trace in the first iteration and then keep it constant.



(a) The original tensor field with a tracked fiber

(b) Smoothing with $\mathbf{J}_{\mathrm{orig}}$ changes the fiber significantly

(c) Weighting $\mathbf{J}_{\mathrm{new}}$ allows to preserve the fiber

Fig. 4. Emphasizing orientation using the new structure tensor can help to preserve the direction of fibers. Here, the glyphs are shaded to indicate the degree of linearity.

Solving Equation (10) with a simple explicit scheme took 30 seconds on the real dataset ($t = 5$) and four seconds in the synthetic case ($t = 250$).

6 Conclusion

We have presented methods for flexible smoothing and segmentation of matrix-valued images. User-controllable options to concentrate on specific features in the high-dimensional data are introduced by integrating Kindlmann's decomposition of the tensor field gradient [12] into the PDE framework of Feddern et al. [3]. We found solutions for cases in which Kindlmann's invariant gradients become ill-defined and proved that the new structure tensor in its unweighted form is equivalent to the previous one. Finally, we successfully demonstrated the advantages of our re-formulation, both on synthetic and on real data.

Future work may use diffusion processes to preprocess DT-MRI data for fiber tracking.

Acknowledgements

We would like to thank Holger Theisel for discussions about the present work.

The human brain dataset[3] is courtesy of Gordon Kindlmann at the Scientific Computing and Imaging Institute, University of Utah, and Andrew Alexander, W. M. Keck Laboratory for Functional Brain Imaging and Behavior, University of Wisconsin, Madison.

Visualization has been done using a modified version of the BioTensor program [21], developed at the University of Utah. Handling the dataset in this framework made it necessary to downsample it by factor two in each dimension.

References

1. Pierpaoli, C., Jezzard, P., Basser, P., Barnett, A., Di Chiro, G.: Diffusion tensor MR imaging of the human brain. Radiology **201** (1996) 637–648
2. Feddern, C., Weickert, J., Burgeth, B.: Level-set methods for tensor-valued images. In Faugeras, O.D., Paragios, N., eds.: Proc. Second IEEE Workshop on Geometric and Level Set Methods in Computer Vision, Nice, France, INRIA (2003) 65–72
3. Feddern, C., Weickert, J., Burgeth, B., Welk, M.: Curvature-driven PDE methods for matrix-valued images. International Journal of Computer Vision **69** (2006) 93–107
4. Chefd'hotel, C., Tschumperlé, D., Deriche, R., Faugeras, O.: Constrained flows of matrix-valued functions: Application to diffusion tensor regularization. In Heyden, A., Sparr, G., Nielsen, M., eds.: Computer vision - ECCV 2002. Volume 2350 of LNCS., Springer (2002) 251–265
5. Welk, M., Feddern, C., Burgeth, B., Weickert, J.: Median filtering of tensor-valued images. In Michaelis, B., Krell, G., eds.: Pattern Recognition. Volume 2781 of LNCS. (2003) 17–24

[3] `http://www.sci.utah.edu/~gk/DTI-data/`

6. Westin, C.F., Knutsson, H.: Tensor field regularization using normalized convolution. In Moreno-Diaz, R., Pichler, F., eds.: Computer Aided Systems Theory - EUROCAST 2003. Volume 2809 of LNCS., Springer (2004) 564–572
7. Rodríguez-Florido, M.A., Ruiz-Alzola, J., Westin, C.F.: DT-MRI regularization using anisotropic tensor field filtering. In: Proc. 2004 IEEE International Symposium on Biomedical Imaging, Arlington, VA, USA (2004) 336–339
8. Wiegell, M.R., Tuch, D., Larsson, H.B., Wedeen, V.J.: Automatic segmentation of thalamic nuclei from diffusion tensor magnetic resonance imaging. NeuroImage **19** (2003) 391–401
9. Rousson, M., Lenglet, C., Deriche, R.: Level set and region based surface propagation for diffusion tensor MRI segmentation. In Sonak, M., Kakadiaris, I.A., Kybic, J., eds.: Computer vision and mathematical methods in medical and biomedical image analysis. Volume 3117 of LNCS., Springer (2004) 123–134
10. Wang, Z., Vemuri, B.C.: An affine invariant tensor dissimilarity measure and its applications to tensor-valued image segmentation. In: Proc. 2004 IEEE Conference on Computer Vision and Pattern Recognition. Volume 1. (2004) 228–233
11. Jonasson, L., Bresson, X., Hagmann, P., Cuisenaire, O., Meuli, R., Thiran, J.P.: White matter fiber tract segmentation in DT-MRI using geometric flows. Medical Image Analysis **9** (2005) 223–236
12. Kindlmann, G.L.: Visualization and Analysis of Diffusion Tensor Fields. PhD thesis, School of Computing, University of Utah (2004) `http://www.cs.utah.edu/research/techreports/2004/pdf/UUCS-04-014.pdf`.
13. Di Zenzo, S.: A note on the gradient of a multi-image. Computer Vision, Graphics, and Image Processing **33** (1986) 116–125
14. Basser, P.J., Pierpaoli, C.: Microstructural and physiological features of tissues elucidated by quantitative-diffusion-tensor MRI. Journal of Magnetic Resonance **B** (1996) 209–219
15. Caselles, V., Kimmel, R., Sapiro, G.: Geodesic active contours. In: Proc. Fifth International Conference on Computer Vision. (1995) 694–699
16. Kichenassamy, S., Kumar, A., Olver, P., Tannenbaum, A., Yezzi, A.: Gradient flows and geometric active contour models. In: Proc. Fifth International Conference on Computer Vision. (1995) 810–815
17. Weickert, J.: Anisotropic Diffusion in Image Processing. Teubner, Stuttgart (1998)
18. Kindlmann, G.L.: Superquadric tensor glyphs. In: Joint Eurographics/IEEE Symposium on Visualization. (2004) 147–154
19. Weickert, J., Kühne, G.: Fast methods for implicit active contour models. In Osher, S., Paragios, N., eds.: Geometric Level Set Methods in Imaging, Vision, and Graphics. Springer, New York (2003) 43–58
20. Sapiro, G.: Vector (self) snakes: a geometric framework for color, texture and multiscale image segmentation. In: Proc. 1996 International Conference on Image Processing. Volume 1. (1996) 817–820
21. BioPSE: Problem Solving Environment for modeling, simulation, image processing, and visualization for biomedical computing applications. Scientific Computing and Imaging Institute (SCI), `http://software.sci.utah.edu/biopse.html`, 2002.

Using Visualizations to Support Design and Debugging in Virtual Reality

Cara Winterbottom, Edwin Blake, and James Gain

Collaborative Visual Computing Laboratory,
Department of Computer Science, University of Cape Town,
Private Bag X3, Rondebosch 7701, South Africa
{cwinterb, edwin, jgain}@cs.uct.ac.za

Abstract. We present a visualization system that helps designers conceptualise interactions in a virtual environment (VE). We use event-condition-action triads (triggersets) for specifying interactions, and provide multiple visualizations: sequence diagrams, floorplans and timelines. We present a two part study: sequencing VE interactions accurately and debugging mistakes. Subjects were divided into two groups: one received visualizations and triggersets and the other (a control group) received triggersets only. The visualization group described 72.5% of the sequence correctly on average, compared to 56.4% by the non-visualization group. The visualization group also detected more than twice as many errors as the control group. The visualization group worked well with multiple, linked windows to create an understanding of the design. Floorplans were most useful for an overview, timelines for understanding specific sequences and sequence diagrams for sequencing and finding mistakes.

1 Introduction

The field of Virtual Reality (VR) has all of the complexities of 3D world creation as well as the difficulties of providing interactions within each world, which are compounded because VR presupposes an independent user. Interactions are the relationships set up between the user of a VE, its objects and the environment itself. The design of interactions is difficult for several reasons: (1) They happen over time for an indeterminate duration, so the design cannot be viewed statically. (2) They happen for various entities, so that each entity or group of entities may participate in a different set of interactions. This leads to a combinatorial escalation of possibilities for interaction. (3) At least some of the interactions will be determined by what the user does, which cannot be pre-specified. Therefore, the designer must deal with a significant amount of uncertainty about how the end result will be experienced. (4) Interactions include actions that are so commonplace to us that we do not naturally think about them, like avoiding obstacles and facing the person to whom we are talking. They require significant detail to define completely. (5) Very often the VE will have a purpose or tell a story, which means that the user must be guided by the interactions and the environment to achieve a goal. (6) The VE must be sufficiently reactive to the user's interactions to make the experience enjoyable and interesting.

In this paper, we describe the use of visualizations to support the design and debugging of interactions within a VE. We accomplish this principally by providing multiple visualizations: a *floorplan*, *timelines* and a *sequence diagram* of the flow of interactions for narrative sequencing. We also describe a study designed to test the effectiveness of these visualizations. In software visualization, which is related to our work, the outstanding questions include: with which problems are diagrams better than text, how do individuals differ in how they work with diagrams and what are the benefits of using multiple representations [1]? These underlie our research and motivate the study. We begin by providing some background in Section 2. In Section 3, our visualizations are described within their context, and in Section 4 we present our study and its results. We conclude in Section 5 with a discussion of the implications of our work.

2 Background

In this section we discuss previous research on program visualization and using visualizations for design and debugging. We also focus on the use of multiple visualizations. Visualization research has a long history, particularly in scientific and information visualization [2,3]. In recent years, research has increasingly been conducted into the use of visualizations for a greater variety of problems [4]. For example, the use of visualizations to support programming tasks, such as debugging and control structure creation [5,6]. Program visualization helps the designer to see the flow of control through a program[5]. To assist in the debugging task, Ko and Myers [7] developed Whyline, which provides visualizations of a program's runtime states in response to questions about what went wrong.

Visualizations can be used by experts during various design processes. General research on external representations suggests that they are useful in promoting reflexivity and a deeper understanding of their subjects [8,1,9,3]. Eastman [9], in a survey of representations used in design, suggests that novices learn from viewing and working with external representations, so that in time these are internalised and become part of the designer's reasoning tools. Petre and Blackwell [10] found that during program design, expert programmers used various forms of mental imagery to think about a task. Commonalities in the imagery used included the fact that they were dynamic, but could be stopped and reversed; they had adjustable granularity; and they included simultaneous multiple images. Baldonado et al. provide guidelines for the use of multiple views in information visualization: they should be used when there is diversity of information, when different views elicit correlations or disparity in the information, or when complex data can be decomposed into manageable chunks. Providing multiple views fosters a deeper understanding of a problem when the distinct representations are understood as describing facets of the same idea [11].

3 A Linked, Multi-view Visualization System

There are various aspects to VE interaction design: the use of space, time and possible sequencing of interactions. These must all be considered in addition to the complexity added by the VE user's freedom of movement. We provide visualizations to help designers understand interactions. The use of multiple visualizations is intended to break up the complexity of interaction design by focusing on different aspects of it. Three visualizations are used: (1) a floorplan visualization of the space of the VE (see the left side of Figure 1a); (2) timelines, which are both a construction and a visualization tool (see Figure 2); (3) a sequence diagram for visualizing and debugging the sequence of interactions (see Figure 1b). The visualizations and how they link together are described further below. For programming the interactions, we use an event-action paradigm. Trigger-condition-action triads, which we refer to as triggersets, are used. The visualization system is embedded in this authoring system. A typical process of using our visualizations is as follows: The designer specifies objects and the environment. Time-based sequences of actions can be entered using timelines. The system generates a floorplan from the object positions and locations in the environment that have been specified. As soon as the designer has entered a few triggersets and object details into the system, she can view the sequence diagram generated by these interactions. Thus, the system allows for incremental programming, as the status and consequences of what has been programmed can be checked at any point by examining the visualizations.

Floorplan. Floorplans have been shown by our own and other experiments to be very useful in the design of 3D worlds [12,13]. They are also used successfully in engineering and architecture to represent space. Floorplans are essentially maps, which allow complex 3D worlds to be viewed in a more simple 2D way [3,14]. In particular, lines of sight, locations and object positions can be easily viewed. In our floorplan, the VE space may be divided into rectangular regions called locations, which can be used in triggers or conditions without specific coordinates. The floorplan automatically displays the positioning of any object that has been given spatial coordinates and indicates its orientation. Any proximity triggers set up in relation to objects are shown on the floorplan. The floorplan is marked with a grid in the units of the world so that distances can be estimated. A compass is used to indicate direction and assist the designer in understanding the rotation system for objects. Designers can interact with the floorplan using direct manipulation. When an object's icon is selected, it is highlighted and the object details are displayed. The floorplan can also be layered to reduce complexity and to show different levels of a 3D world, if necessary. Figure 1a is an example of a floorplan used in the system.

Sequence Diagram. Our sequence diagrams are inspired by Harel's statecharts [15], which were developed for designing complex reactive systems. They are generated directly from the triggersets and follow the flow of data through the VE. States are identified where specific interaction possibilities exist. Each state specifies the current conditions in the environment that allow triggersets to execute. The states are linked by arrows, which correspond to triggersets' execution. If a triggerset does not change the current interaction possibilities, its arrow leads back to the same state. When the user clicks on a state, the

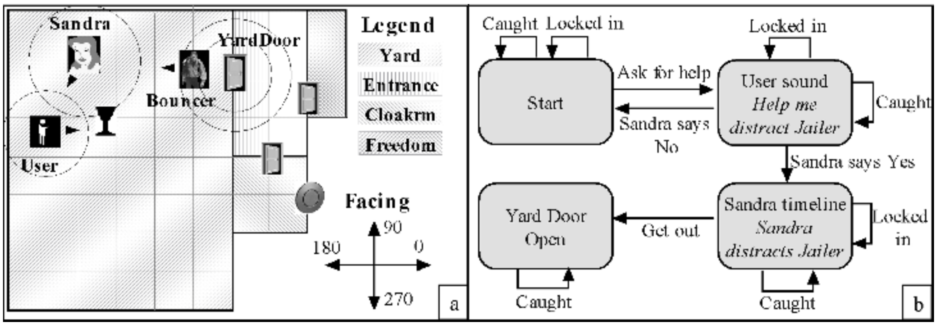


Fig. 1. Floorplan and Sequence Diagram. (a) is a floorplan of a VE. The position and orientation of each object is indicated by an icon and directional arrow, respectively; circles around each object correspond to proximity triggers that have been set up. (b) is a corresponding sequence diagram showing states and links. A state is a point from which user interactions will have consequences, and a link is a triggerset that can be executed from its originating state.

arrows that leave it and the states to which they connect are highlighted. A description of the state also pops up. When the user clicks on an arrow, all other arrows referring to the same triggerset are highlighted and a description of the triggerset pops up. In this way, designers can step through their interactions and think about how the sequence fits together. For debugging interactions, the sequence diagram allows designers to see the effects of the triggersets; where triggersets have unexpected consequences and which triggersets never execute. A sparse diagram will indicate a lack of interactions provided in the VE. Because the sequence diagram itself is not linear, designers are encouraged to view the interactions in a non-linear way. Figure 1b provides an example of a sequence diagram.

Timelines. Timelines are a well understood formalism and have been shown to reduce errors in temporal ordering [11,12]. Because VE authoring depends on user interactions, our timelines do not represent VE time from start to finish. Instead, they represent parts of the VE where a sequence of actions will happen in known time. The actions are grouped on the timeline and then treated as a single action. For example, if a story is told in a VE, the sound file and actions of storyteller and listeners must all be coordinated. The timeline can be used to sequence the storytelling and reactions with precision and efficiency. The story timeline can then be executed as a single action. Each row on the timeline corresponds to an object and contains any actions that the object performs. The length of the action is automatically calculated (e.g., an animation's basic length is read from its file and then multiplied by repetitions) and visualized on the timeline. Any action on a timeline can be selected to uncover more details about it. Figure 2 displays a typical timeline from the system.

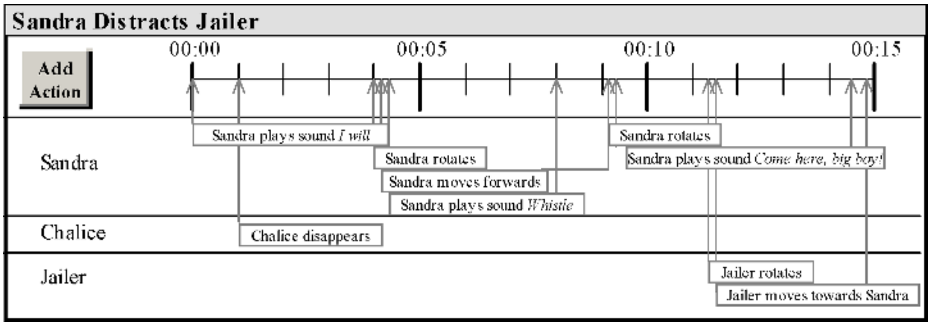


Fig. 2. Timeline showing objects involved and their actions. These can be selected to elicit more detail about each action.

Linking Between Visualizations. All our visualizations are linked to each other. The objects and locations referred to in each visualization connect them to one other and to the triggersets, so that they can be cross-referenced. Timelines are described in the sequence diagram. For example, Figure 1b displays a sequence diagram for a VE where the state, *Sandra Timeline Sandra Distracts Jailer*, describes the state of the VE while the timeline, *Sandra Distracts Jailer* is playing. From this diagram, one can see that the triggerset *Sandra Says Yes* begins the timeline and that three triggersets can be activated by the timeline or actions contained in it. If the designer selects this state, a detailed description of the timeline is provided and if the designer selects any of the triggersets connected to the state, a detailed description of each is given. The designer can use the name of the state to open the actual timeline (Figure 2). The names of the links can be used in a similar way to find the actual triggersets to which they correspond. The names of objects and locations in the sequence diagram, timelines and triggersets can also be used to find objects on the floorplan (Figure 1a). Therefore, by working through the visualizations in combination, the designer can gain an overview of the sequence of triggersets and how they interact with each other, as well as a detailed description of each.

4 Visualization Study

An exploratory study was conducted to test the effectiveness of our visualizations. For this initial study we wanted to find out how effectively the visualizations and triggersets were used by people in understanding a sequence of 3D interactions and in debugging errors in a sequence. This aim has four parts: (1) To assess how subjects were able to describe the possible sequence of interactions in a VE. (2) To investigate the extent to which subjects could identify errors in a second set of interactions. (3) To investigate which visualizations subjects preferred to use in different contexts. (4) To assess how well subjects work with multiple visualization windows. We included a control group, who did not receive visualizations, so that we could compare the performance of subjects who did receive visualizations with that of subjects who did not. Because of the exploratory nature of our work, we decided to use participant observation for our study [16]. We observed subjects working on simple problems using our system. After the tasks were finished, the researcher discussed the subjects' experiences with them in a structured interview.

Part of the study dealt with identifying problematic interactions. Most mistakes, in our experience, fall into one or more of four general categories: *timing* errors arise from the time it takes the user or other objects to complete actions; *spatial* errors arise from the way space is used, in terms of orientation and location of objects; *sequencing/logical* errors arise from problematic ordering of interactions and the way they interrelate; and *implicit assumption* errors arise from the designer forgetting to state all behaviour explicitly. Examples of these errors are shown in Table 1.

4.1 Description of the Study

Our study had two parts, each in a separate VE. Visualization subjects were provided with a floorplan, timeline(s) and a sequence diagram of each. For the Sequence part of the study, a simple VE entitled *Bouncer Example* was used. The physical space of the *Bouncer Example* consisted of three locations. In addition to the User avatar, the VE contained a Bouncer (or guard), a Door, a Bell and a Chalice. The goal was to distract the Bouncer away from the Door that he was guarding, so that the user could open the door and grab the chalice. Various triggersets were set up to describe the possible interactions. The aim of this part of the study was to see how well subjects could work out what might happen in the VE. Therefore, the triggersets were set up so that some could not execute without others having been triggered, so that there was a sequence.

For the Debug part of the study, a VE entitled *Jail Example* was conceptualised. The space consisted of four locations. In addition to the User avatar, the VE contained a Jailer, an object named Sandra, three Doors, a Push-Button and a Chalice. The goal of the VE was to use Sandra to distract the Jailer, so that the User could get into the location named Freedom. The floorplan and sequence diagram for this VE are shown in Figure 1. The *Sandra Distracts Jailer* timeline shown in Figure 2 is also from this VE. As in the Sequence part of the study, various triggersets had been set up describing the interactions that could happen. However, in this case, several mistakes had been introduced into the triggersets. The aim here was for each subject to identify as many potential errors as possible. This was intended to test how well subjects could debug incorrectly programmed interactions. The mistakes and the error categories from which they draw are indicated in Table 1.

4.2 Procedure

A sample of eleven graduate students from different disciplines was recruited. It was decided to use people with graduate degrees or equivalent working experience, as these corresponded to the target group of people who might work with VR. None of the subjects had any experience with graphics programming,

Table 1. Interaction Programming Mistakes Introduced into the Visualization Study

Mistake	Description	Error Category
Jmove	Jailer does not move far enough and so proximity trigger is not executed	Spatial/Timing
Fopen	No way specified to open a door behind which is a button to open the door to Freedom location	Implicit Assumption / Spatial
CLcon	Caught and Locked In triggersets both execute and conflict	Sequencing and Logical
YNcon	Sandra Yes and Sandra No triggersets both execute and conflict	Sequencing and Logical

although they had experience with graphics packages ranging from none to Illustrator and 3D Studio Max. Gender was evenly represented. Subjects were volunteers within these constraints and were paid a small amount for their participation. A control group of five subjects who would not receive visualizations was randomly selected from the larger sample.

Each subject experienced the study individually and was then interviewed. Before the study began, the subject was given an introduction in which interactions, the triggerset formalism and the visualizations (only for the visualization group) were introduced. Following this, subjects were given 20 minutes to examine the triggersets and visualizations for the Sequence part of the study. After this they were asked to write down what might happen in the VE, indicating any dependencies among the triggersets. Thereafter, the instructions and VE descriptions for the Debug part of the study were provided. Subjects were asked to specify any mistakes that they found and were also given 20 minutes to examine the triggersets for this part. When subjects had finished with both parts of the study, a structured interview was conducted. A code was developed for scoring the accuracy of the sequence descriptions of the Sequence part of the study. Points were given for correct sequencing, awareness of branches, awareness of object locations and how these might change, and awareness of pre-conditions on actions. An independent marker examined the scripts using the code, in order to ensure that the marking was unbiased.

4.3 Results and Discussion

All of the subjects found the triggerset formalism easy to use and understood how the triggersets worked. It was when they had to be ordered in a sequence that subjects had difficulties. Visualization subjects achieved a 72.5% average on correctly working out the Sequence part of the study, while the control group only achieved a 54.6% average. The different between means was just not significant ($F = 6.32,\ p < 0.09$). However, once an outlier in the control group was removed, its average dropped to 48.5% and the difference became significant ($F = 1.43,\ p < 0.002$). For the Debug part of the study, the visualization group noted twice as many errors as the control group (37.5% vs. 15%), but overall the number of errors noted was low. This result was not tested for significance, as most of the control group(3 of 5)found zero errors, which skewed their results. Subjects were limited to 20 minutes to debug the VE, which partially accounts for the poor results. Only one (visualization) subject found Jmove and CLcon (see Table 1 for a description of the errors). Two visualization subjects and one control subject found Fopen. But five visualization subjects (and two control subjects) found YNcon. We can examine the errors that were detected, broken into error types. The visualizations seem most helpful for sequencing errors, probably because the sequence diagram indicates possible sequences. Without this, the triggersets must be manually connected to each other, based on the result of each one being executed. Timing errors are almost impossible to find without some way to view the final product or step through the events.

Subjects found the floorplan most useful: to orient, give a concrete sense of the space and where the objects are in relation to each other: "Once you coordinate between the physical locations, you can see how you need to move." In fact, half of the subjects stated that they could not have reconstructed the sequence without the floorplan. Timelines were useful for noticing a predictable sequence: "Used the timeline for Bouncer Move to see how he went away." One subject stated, "I did not use timelines much to examine interactions because they were simple, but they would be very useful to make actions — work very nicely. For design, I like the timeline. It is important as both a visualization and a construction tool." Mistakes in the Debug part of the study were made more obvious by the sequence diagram: "The sequence diagram is useful for seeing how the triggersets relate, their order and what activates what." "Could walk anywhere, but the VE only reacts like the sequence." Even those who used the sequence diagram less stated that it was useful to "check up after your own analysis of the triggers". Subjects did state that they wanted more interactivity from the sequence diagram.

A typical method of working with the visualizations was to begin with the floorplan in order to gain an overview of the VE and its interactions. Subjects would then examine the triggersets and object descriptions, referencing the names and positions of objects and locations on the floorplan. During this process they would repeatedly go to the sequence diagram to find out how triggersets sequenced, or confirm their own analyses. When timelines were referred to, they would open them and check the objects and actions. In this way, subjects used the linking between the visualizations effectively. Subjects all worked in different ways with the visualizations and triggersets. This justifies the flexibility of the tool in allowing for different work processes. There were similarities, though: Subjects all looked at multiple visualizations and most cross-referenced the visualizations constantly to work out what was happening in the VE. Only one subject mentioned problems with switching between multiple windows. They all found the visualizations clear and useful but found each one useful for different things, e.g. "I had all three (visualizations) open at the same time. Then, if you don't understand the sequence you can look at the timeline." And "The triggersets are basically just the details of the rest (the visualizations)". They all felt that the tasks would have been much harder without the visualizations. A few mentioned that the interactivity was very helpful.

Our study also indicated areas where we could improve the visualizations. Subjects underused the timelines, as they were more difficult to access. This means that *ways to access the visualizations must be made highly visible and simple.* People need to be able to access each visualization from the others, so that they do not overlook them and to make it easier to cross-reference. Subjects did not often drill down into triggersets and object details to find out necessary information, e.g., the angle of rotation for an accurate point of view. Therefore, *details that are not visualized must be made more obvious and easily accessible.* Subjects indicated that it would be helpful to group triggersets according to various criteria, such as locations in which they might execute. Users also need more help with working out the consequences of rotations and translations, in terms of where an object will end up after a sequence. *More support is needed to help users understand certain parts of 3D interaction design, such as spatial relationships.* The visualizations can guide users more in this regard.

5 Conclusion

We have described our experiences in providing visualizations to support the understanding of VE interaction design. In the study we conducted, visualization subjects consistently out-performed the control group. They understood the visualizations and worked well with them. They naturally used multiple interacting visualizations to build a complete idea of a complex design. Floorplans were used to gain an overview of the VE and interactions, timelines were used to understand the consequences of predictable sequences of events, and sequence diagrams were used to understand how the interactions worked together and to find mistakes. The interactivity that was added was also very positively received, although subjects would have preferred more. More work must be done on assisting users with debugging tasks, especially in terms of highlighting inappropriate or inconsistent interaction effects.

These results show that using visualizations to understand and debug VE interactions improves the authoring process. This is a key finding in helping to make VR more accessible as a creative medium. It also provides us with validation of the efficacy of our visualizations, which creates a solid basis for future work. Our next step is to add a 3D view of the VE to complement the other visualizations. Then we will develop a run mode with a highlight which moves through all of the visualizations, indicating where the action is taking place. More interactivity will also be added between the 3D window and the other visualizations.

Acknowledgements

This research is funded by the South African National Research Foundation (GUN 2053402) and by the University of Cape Town.

References

1. Blackwell, A., Whitley, K., Good, J., Petre, M.: Cognitive factors in programming with diagrams. Artificial Intelligence Review **15** (2001) 95–113
2. Card, S., Mackinlay, J., Shneiderman, B.E.: Readings in Information Visualization: Using Vision to Think. Morgan Kaufmann Publishers, California (1999)
3. Shu, N.: Visual Programming. Van Nostrand Reinhold Company, New York (1988)
4. van Wijk, J.: The value of visualization. In Silva, C., Groeller, E., Rushmeier, H., eds.: Proceedings of IEEE Visualization, IEEE (2005) 79–86
5. Romero, P., Cox, R., du Boulay, R., Lutz, R.: A survey of external representations employed in object-oriented programming environments. Journal of Visual Languages and Computing **14** (2003) 387–419
6. Myers, B.: Taxonomies of visual programming and program visualization. Journal of Visual Languages and Computing **1** (1990) 97–123
7. Ko, A., Myers, B.: Designing the whyline: A debugging interface for asking questions about program behaviour. In: Proceedings of CHI 2004. (2004) 151–158
8. Kavakli, M., Suwa, M., Gero, J., Purcell, T.: Sketching interpretation in novice and expert designers. In Gero, J., Tversky, B., eds.: Visual and Spatial Reasoning in Design, Australia, Key Centre of Design Computing and Cognition (1999) 209–220
9. Eastman, C.: New directions in design cognition: studies of representation and recall. In Eastman, C., McCracken, M., Newsletter, W., eds.: Design Knowing and Learning: Cognition in Design Education, Elsevier Science (2000)
10. Petre, M., Blackwell, A.: Mental imagery in program design and visual programming. International Journal of Human-Computer Studies **51** (1999) 7–30
11. Baldonado, M., Woodruff, A., Kuchinsky, A.: Guidelines for using multiple views in information visualization. In: Proceedings of AVI, ACM Press (2000)
12. Harada, K., Tanaka, E., Ogawa, R., Hara, Y.: Anecdote: A multimedia storyboarding system with seamless authoring support. In: ACM Multimedia, ACM Press (1996) 341–351
13. Coleburne, A., Rodden, T., Palfreyman, K.: VR-MOG: A toolkit for building shared virtual worlds. In Slater, M., ed.: Proceedings of FIVE (Framework for immersive virtual environments) Working Group Conference. (1995) 109–122
14. Tufte, E.: The Visual Display of Quantitative Information. Graphics Press, Cheshire, UK (1983)
15. Harel, D.: On visual formalisms. Communications of the ACM **31** (1988) 514–530
16. Banister, P., Burman, E., Parker, I., Taylor, M., Tindall, C.: Qualitative Methods in Psychology: A Research Guide. Open University Press, Buckingham, UK (1994)

Strategies for Part-Based Shape Analysis Using Skeletons

Wooi-Boon Goh

School of Computer Engineering, Nanyang Technological University,
Nanyang Avenue, Singapore 639798
aswbgoh@ntu.edu.sg

Abstract. Skeletons are often used as a framework for part-based shape analysis. This paper describes some useful strategies that can be employed to improve the performance of such shape matching algorithms. Four key strategies are proposed. The first is to incorporate ligature-sensitive information into the part decomposition and shape matching processes. The second is to treat part decomposition as a dynamic process in which the selection of the final decomposition of a shape is deferred until the shape matching stage. The third is the need to combine both local and global measures when computing shape dissimilarity. Finally, curvature error between skeletal segments must be weighted by the limb-width profile along the skeleton. Experimental results show that the incorporation of these strategies significantly improves the retrieval accuracy when applied to LEMS's 99 and 216 silhouette database [10].

1 Introduction

Medial or skeleton-based approaches to shape analysis have been widely used and proven to be very effective in representing and classifying 2D shapes [4], [12], [13]. Additionally, a recent paper by Kimia [9] provides a comprehensive discussion of the possible role of medial geometry in the human visual system. Neurophysiological experiments observe peak neural activity in the V1 cells at localities of medial axis and psychophysical experiments suggest that we may use medial points when encoding shape information. We have been adopting a medial-based framework in our attempt to develop robust algorithms for retrieving and classifying 2D silhouettes from shape databases. During the course of our research, several very important strategies were discovered that allowed us to improve the robustness of our shape-based image retrieval applications. This paper presents four of these strategies and described how they are implemented with the multiresolution gradient vector field (MGVF) skeleton framework presented in [5], [7]. An important feature of the MGVF framework that is relevant to this work is the availability of a vector field directional disparity map $M_R(x,y)$ given by equation (6) in [5]. The paper will also describe how this readily available disparity measure can be utilized in the implementation of the proposed strategies. Four strategies will be discussed in section 2 and experimental results will be presented in section 3.

2 Strategies for Robust Part-Based Shape Analysis

2.1 Incorporation of Ligature-Sensitive Information

Unstable ligatures and their effects on skeletal topology and visual saliency must be explicitly incorporated into the shape analysis process. Ligatures [1] are problematic because they can appear and disappear with non-visually salient changes in the shape. If not properly handled, such unstable skeletal features can alter skeletal topology and also skew the matching cost between visually similar shapes. Since the MGVF skeleton is used, a visual saliency measure based on the average vector field directional disparity measure $M_R(x,y)$ computed along the skeleton is proposed. This disparity-based saliency correlates well with the measure of ligature stability (see Fig. 1a). Other measure for ligatures such as in [14] can also be used. During shape matching, the disparity saliency associated with each skeletal part is multiplied with its corresponding part matching cost to appropriately weigh the contribution of the part's mismatch to the overall shape dissimilarity (see Fig. 1b). Ligatures should also be taken into account during part decomposition, as is discussed in the next section.

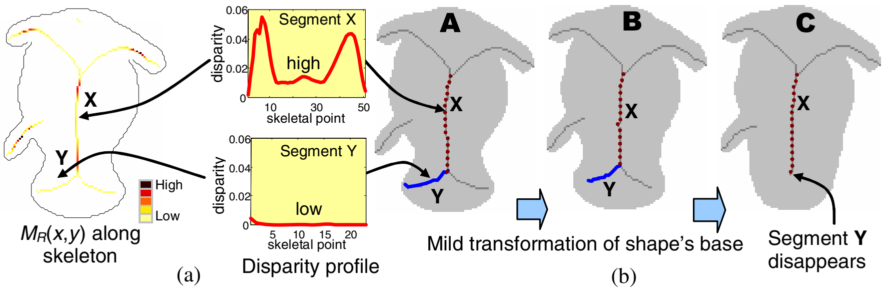
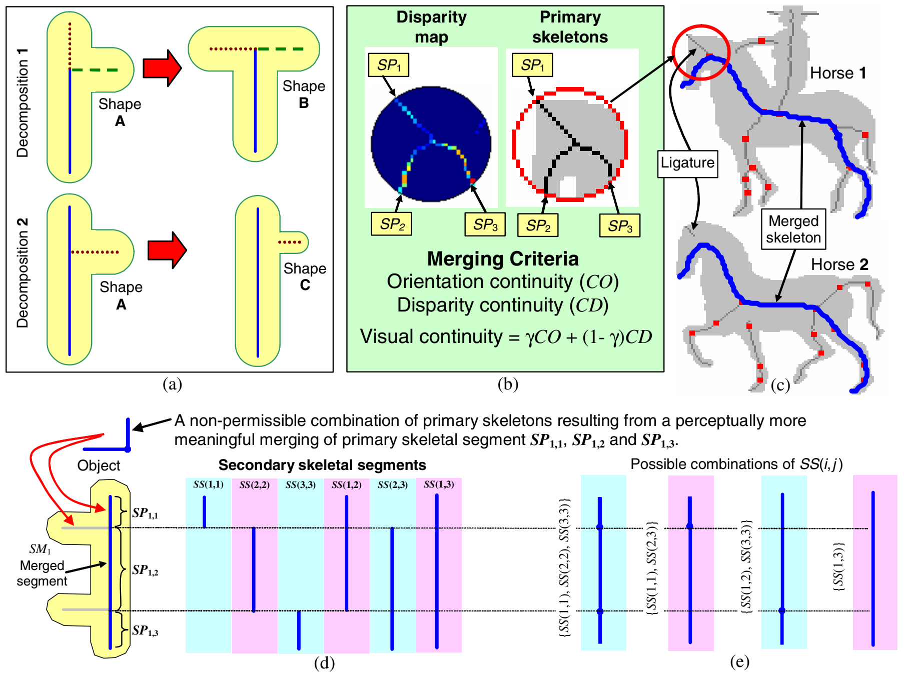


Fig. 1. (a) Unstable ligatures are associated with the low disparity values $M_R(x,y)$ along skeletal segment (e.g. segment Y). (b) The application of the weak disparity saliency measure will reduce the mismatch cost associated with missing segment **Y** when matching shapes **A** and **C**.

2.2 Dynamic Part Decomposition

Traditionally, a shape's skeleton is first decomposed into parts based of some set of rules [2], [11]. Dissimilarity between two shapes is then computed by finding the optimal combination of matches between these parts. However, part decomposition is a dynamic and adaptive process that depends on the two shapes being compared. Fig. 2a shows that the intuitive decomposition of shape **A** is different when it is forced to match different shapes **B** and **C**. In order to implement dynamic part decomposition, the skeleton of a shape is segmented into *primary* skeletal segments using various segmentation criteria such as skeletal junction, skeletal curvature maxima-minima, etc. A merging process is then employed to re-group these segmented pieces into visually continuous and coherent *merged segments*. The merging process follows after the notion of *visual conductance* that Katz and Pizer [8] used to derive natural parts-hierarchy. Like in [8], orientation continuity is adopted but more importantly, we also introduced another merging criterion based on the similarity of mean disparity values (obtained from $M_R(x,y)$ in [5]) of each skeletal segment at the locality of the skeletal junction. The disparity continuity criterion is important as it prevents unstable ligatures from interfering with the decomposition process (see Fig. 2b). Notice in Fig. 2c, the principal components of the both horses' structural skeletons were consistently re-grouped despite initial separation due to ligatures.



Decomposition 1
Decomposition 2
Shape A
Shape B
Shape A
Shape C
(a)
Disparity map
Primary skeletons
SP1
SP1
SP2
SP3
SP2
SP3
Merging Criteria
Orientation continuity (CO)
Disparity continuity (CD)
Visual continuity = γCO + (1- γ)CD
(b)
Horse 1
Ligature
Merged skeleton
Horse 2
(c)
A non-permissible combination of primary skeletons resulting from a perceptually more meaningful merging of primary skeletal segment SP1,1, SP1,2 and SP1,3.
Object
SM1
Merged segment
SP1,1
SP1,2
SP1,3
Secondary skeletal segments
SS(1,1)
SS(2,2)
SS(3,3)
SS(1,2)
SS(2,3)
SS(1,3)
(d)
Possible combinations of SS(i,j)
{SS(1,1), SS(2,2), SS(3,3)}
{SS(1,1), SS(2,3)}
{SS(1,2), SS(3,3)}
{SS(1,3)}
(e)

Fig. 2. (a) Part decomposition is dependent on the shape being matched. (b) Skeleton merging at Horse **1**'s head highlights how using disparity continuity (*CD*) avoids topological interference by ligatures despite the strong orientation continuity (*CO*) at the junction [7]. (c) Appropriate disparity continuity emphasis (using $\gamma=0.1$) results in similar merged skeletons for Horses **1** and **2**. (d) All possible secondary skeletons extracted from an ordered set of 3 merged primary skeletons and (e) their possible combinations to reconstitute the merged segment.

A merged segment can be decomposed into various *secondary* skeletal segments (see Fig. 2d) and the permutations of these *secondary* segments that reconstructs the merged skeleton describe all permitted part decomposition options available for that merged segment (see Fig. 2e). The preferred option is not decided until during shape matching when similar information regarding the shape being matched is also available (see Fig. 3). Notice that appropriate merging of segmented skeletons serves to reduce the large combinatorial possibility of part decomposition. And a ligature-sensitive visual continuity criterion encourages the selection of decomposition permutations that are visually consistent over different shapes that share similar gross visual forms. Dynamic decomposition is similar in principle to graph topology altering algorithms such as [4], [12], [13] that use insert, merge, contract operations to accommodate deformation arising from occlusion and limb growth.

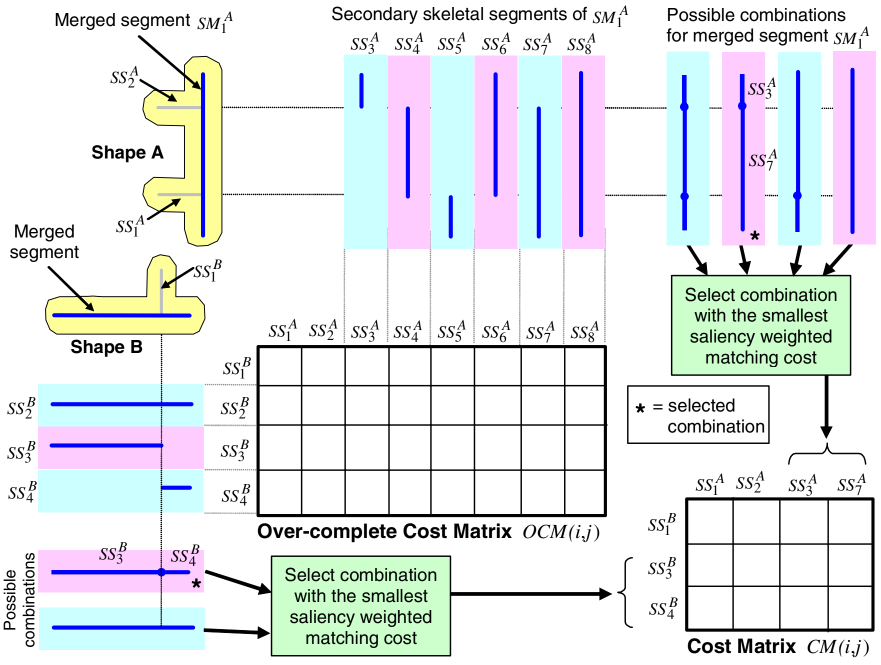

Merged segment SM_1^A
Secondary skeletal segments of SM_1^A
Possible combinations for merged segment SM_1^A
SS_2^A
Shape A
SS_1^A
Merged segment
SS_1^B
Shape B
Select combination with the smallest saliency weighted matching cost
* = selected combination
Over-complete Cost Matrix $OCM(i,j)$
Possible combinations
Cost Matrix $CM(i,j)$

Fig. 3. Overview of the shape matching algorithm. An over-complete saliency-weighted cost matrix $OCM(i,j)$ is created from the sum of part and global distances (see section 2.3) computed for every combination of secondary skeletons between shapes **A** and **B**. The least cost decomposition option for each merged skeleton is identified, leading to a cost matrix $CM(i,j)$ that contains no repetition in skeletal segments. The Hungarian algorithm with appropriate dummy cost weighting as in [3] is used to identify the part match combinations between shapes **A** and **B** that yield the lowest total part matching cost. Further details can be found in [7].

2.3 Combination of Both Local and Global Shape Measures

Part descriptors employing only local features cannot capture the coherent structure of a shape. Based on local measures alone, dissimilar shapes **A** and **B** in Fig. 4a are considered good matches as all parts in one have a local representation in the other. However, we can easily tell apart the two categories of spectacles and dumb-bells in Fig. 4b even though they share similar local parts. The observed categorization can only be realized by incorporating global measures that take into account the global part relationships within the shape. But purely global measures have drawbacks as they are sensitive to limb articulation (see Fig. 4c). A robust shape dissimilarity measure must combine both *part distance* (local) and *global distance* measures in adjustable proportions depending on the application or data set at hand (see Fig. 7g and 7h). In this work, the global distance is computed using a combination of a spatial and two angular relationships between skeletal segments and a reference point (see Fig. 4d and 4e). The part distance, on the other hand, is obtained by matching a combination of local part descriptors comprising of orientation histograms [6], [7] skeletal

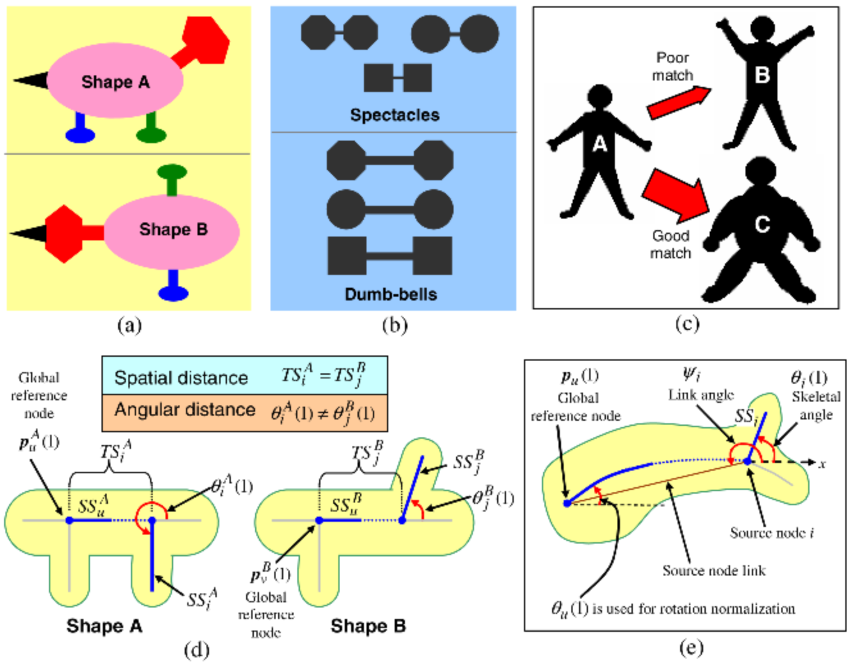


Fig. 4. (a) Two dissimilar shapes that share identical local part features. (b) Global geometry help group shapes sharing similar parts into different categories. (c) Part articulation is problematic for shape descriptors using only global geometry. (d) Shapes (**A** and **B**) whose limbs differ in their angular but not in their spatial global relationships. (e) The global angular distances used consist of the source node skeletal angle $\theta_i(1)$ and link angle ψ_i that are measured relative to a selected global reference node. They are rotation-normalized by the skeletal angle $\theta_u(1)$ at the global reference node (see Appendix for details on finding global references).

curvature and limb width profile. Implementation details are beyond the scope of this paper but using skeletal curvature requires the introduction of the next strategy.

2.4 Application of Curvature Saliency to Curvature-Based Part Matching

Fig. 5a and 5b suggest that the mismatch in skeletal curvature does not necessarily mean two shapes are dissimilar, implying that a part's *curvature-based visual saliency* is not constant along its skeleton. This saliency variation is essentially due to the limb's width along the skeleton (i.e. radius of the maximal disk centered about a skeletal point). A perceptually meaningful measure of the skeletal curvature error between skeletons should incorporate curvature saliency weighting along its length (see Fig. 5c). A global normalization factor (e.g. square-root of query shape's area) should be employed to ensure the limb-width weighting applied to the skeletal curvature error is proportional to the shape's size and not the current skeletal segment.

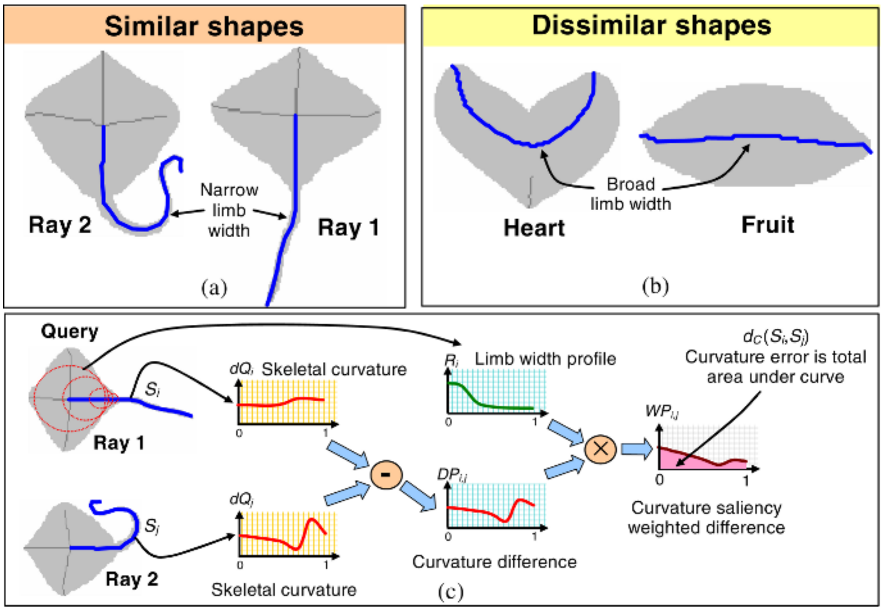


Fig. 5. Contribution of skeletal curvature to shape dissimilarity is dependent on the limb width profile along the skeleton. (a) Two similar shapes with dissimilar skeletal curvature. (b) Two dissimilar shapes that are differentiate by their differing skeletal curvature. (c) The computed skeletal curvature error between skeletons is multiplied by the limb-width variation along the query shape's skeleton in order to incorporate curvature saliency weighting.

3 Experimental Results

3.1 Silhouette Classification

The proposed part-based shape analysis (PBSA) strategies that were implemented within the MGVF skeleton framework of [5] was tested on the 99-shape silhouette database used in [12] (available online at [10]). As can be seen in Fig. 6a, this is a difficult data set to classify due to the presence of large within-class variations (*four-footed mammal*, *greeble*), limb articulation (*man*, *hand*) and occlusions (*hare*, *plane*). Results using the proposed algorithm (labeled MGVF-PBSA) are summarized in Fig. 6b with a precision-recall diagram similar to that used in [12]. The incorporation of the proposed strategies produced retrieval results that are better than those using Sebastian *et. al*'s shock graph edit-distance algorithm [12], which is probably the current state-of-the-art in 2D skeleton-based shape analysis. Comparison with shape context [3] is not entirely fair (it is not a part-based descriptor) but is included to show the need for an adaptive part-based decomposition strategy when dealing with shape databases featuring significant occlusion and limb articulation.

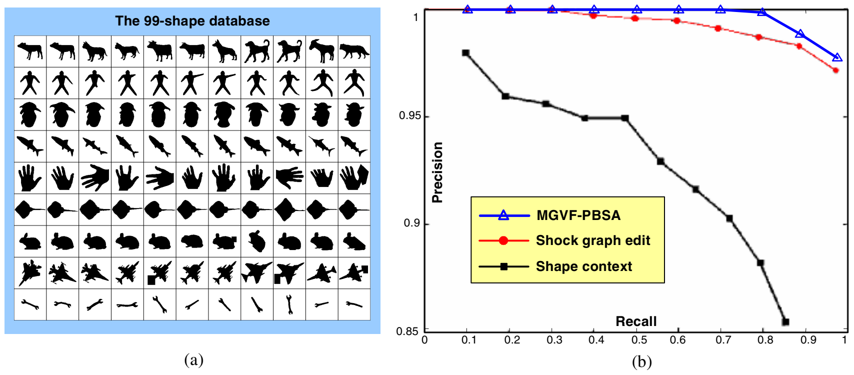


Fig. 6. (a) The shapes (arranged in their respective categories) in the 99-shape database. (b) The precision-recall diagram showing the improved shape-based retrieval performance of the proposed algorithm (MGVF-PBSA) compared to the shock graph edit-distance of [12] and the shape context of [3], both of whose results on the 99-shape database were replicated from [12].

3.2 The Influence of Incorporating Ligature-Sensitive Saliency Weighting

Fig. 7a shows a rapid fall-off in retrieval accuracy when skeletal segments were given equal weightage during the computation of the shape distance regardless of its formation stability. This result supports the fact that medial-based shape analysis algorithms must explicitly incorporate ligature-sensitive information to weigh the influence of part mismatches during shape matching. Mismatches observed in Fig. 7c suggest that the stable structural backbone of the *greeble* shape (which is straight) should be given more weightage during part-based shape matching so that it can be distinguished from very similar *hare* shapes (whose backbone is curved). Especially in the presence of large within-class variations in the form of different-shaped ‘ears’ and ‘limbs’.

3.3 The Influence of Incorporating Global Distance

The shape retrieval performance in Fig. 7d degraded when inter-part spatial and angular relationships (global distances) were not taken into account during shape matching. The nature of the mismatches is clearly observed in Fig. 7f when only local part descriptors were employed. Many *hand* shapes were incorrectly retrieved as their fingers found good local part matches with the *mammals*’ legs. This result supports the use of global distance especially if large within-class variations exist in the data set. However, excessive global emphasis in not always favorable, especially if the objects within the same category have articulating parts (see Fig 7g). As global distance emphasis was reduced, perfect retrieval results were then obtained (see Fig. 7h).

3.4 Incorporating Curvature Saliency Weighting

Sample retrieval results in Fig. 8 highlights the importance of using the limb-width profile to weigh the skeletal curvature errors computed along the skeleton. Notice in Fig. 8b, the large curvature variations in the *elephants*’ trunks and the *rays*’ tails increased misclassifications in the absence of curvature saliency weighting.

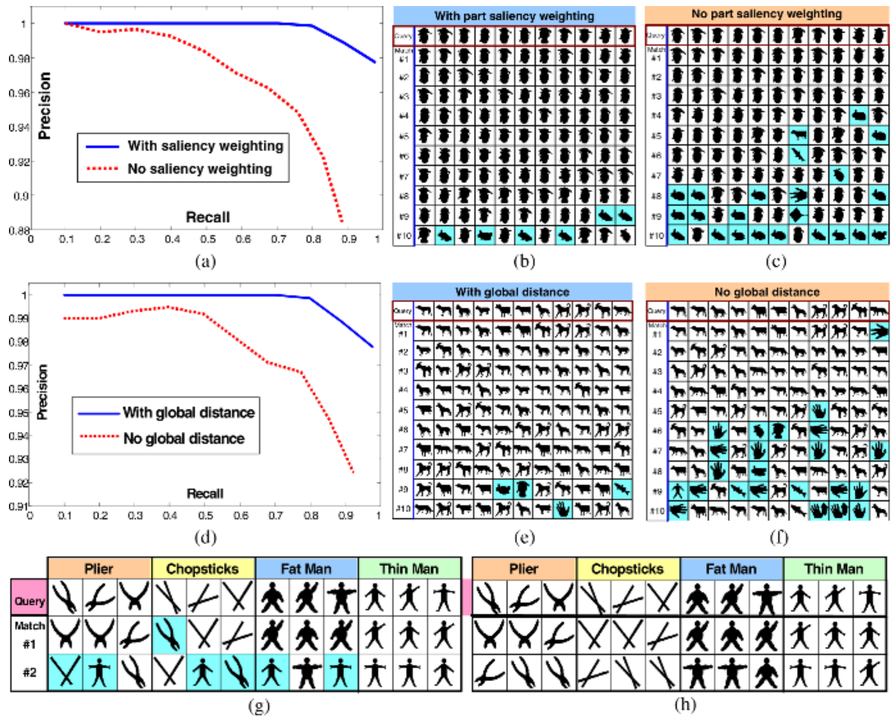


Fig. 7. (a) Precision-recall diagram showing the positive influence of using disparity-based saliency during shape matching. Sample retrieval results for the *greeble* category are shown (b) with and (c) without using disparity saliency weighting. For each query shape (topmost), the 10 best shape matches from the 99-shape database are shown ordered from #1 (best) to #10. The retrieval of the query shape itself is excluded from the display. Incorrect classifications are shown shaded. (d) Precision-recall diagram showing the positive influence of using global geometry during part-based shape matching. Sample retrieval results for the *4-footed mammal* category are shown (e) with and (f) without using global geometric information. (g) Ordered classification results obtained for a data set of articulating objects and another when (h) the global distance contribution was reduced by a factor of 0.3 relative to the part distance.

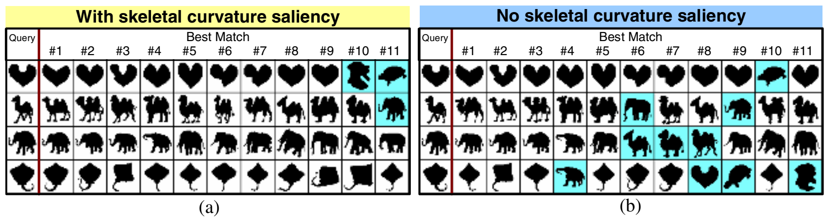

With skeletal curvature saliency
No skeletal curvature saliency
Query
Best Match
#1 #2 #3 #4 #5 #6 #7 #8 #9 #10 #11
(a)
(b)

Fig. 8. Sample retrieval results from the 216-shape database [10] (a) with and (b) without curvature saliency weighting

4 Conclusions

Four strategies were proposed to improve the performance of skeleton-based shape analysis algorithms. Firstly, ligatures in skeletons must be explicitly accounted for in both shape matching and part decomposition. Additionally, part decomposition must be adaptive to the shape that is being compared to. In our implementation, this entailed the computation of all visual coherent part decomposition options for a shape using skeletal segmentation and subsequent re-merging based on criteria that took into account ligatures. The selection of the decomposition option was done during shape matching. Both local and global measures must be used to compute shape dissimilarity and skeletal curvature matching, if used, must be appropriately weighted with the skeleton's limb-width profile. Emphasis between local and global distances is dependent on the nature of the shape matching application and data set. Implemented within the MGVF skeleton framework of [5], the experimental results showed that adopting these strategies produced improved shape retrieval performances. Application of these strategies on other skeleton-based shape analysis algorithms should yield similar benefits, albeit their implementations may be realized differently.

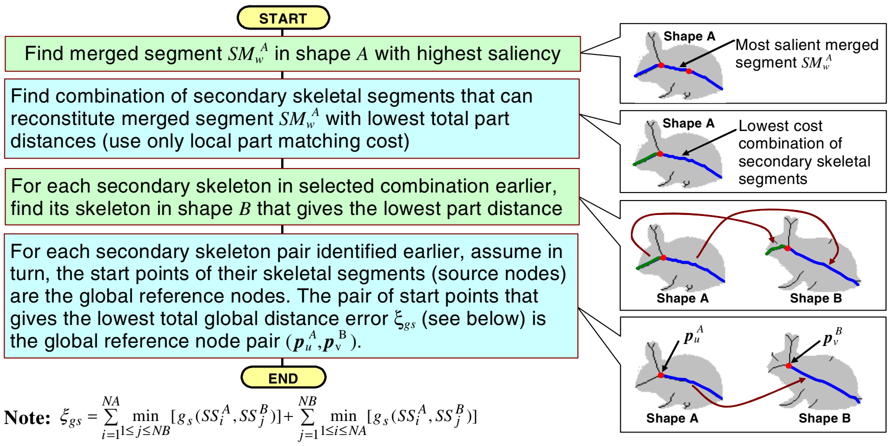
References

1. August, J., Siddiqi, K.S., Zucker, S.W.: Ligature Instabilities in the Perceptual Organization of Shape. Proc. of the Conf. on Computer Vision and Pattern Recognition 2 (1999) 42-48
2. di Baja, G.S., Thiel, E.: (3,4)-weighted Skeleton Decomposition for Pattern Representation and Description. Pattern Recognition, 27 (8) (1994) 1039-1049
3. Belongie, S., Malik, J., Puzicha, J.: Shape Matching and Object Recognition using Shape Contexts. IEEE Trans. on Patt. Analysis and Machine Intelligence 24 (24) (2002) 509-522
4. Geiger, D., Liu, T.L., Kohn, R.V.: Representation of Self-Similarity of Shapes. IEEE Trans. on Pattern Analysis and Machine Intelligence 25 (1) (2003) 86-99
5. Goh, W.B., Chan, K.Y.: Structural and Textural Skeletons for Noisy Shapes. In: Bebis, G., Boyle, R., Koracin, D., Parvin, B. (eds.): Advances in Visual Computing. Lecture Notes in Computer Science, Vol. 3804. Springer-Verlag, Berlin Heidelberg (2005) 454–461.
6. Goh, W.B., Chan, K.Y.: Part-based Shape Recognition using Gradient Vector Field Histograms. In: Petkov, N., Westenberg, M.A. (eds.): Computer Analysis of Images and Pattern. Lecture Notes in Computer Science, Vol. 2756. Springer-Verlag, (2003) 402–409.
7. Goh, W.B.: Shape Analysis using Multiresolution Gradient Vector Field. Ph.D. Thesis, Nanyang Technological University, Singapore (2005).
8. Katz, R.A., Pizer, S.M.: Untangling the Blum Medial Axis Transform. International Journal of Computer Vision 5 (2/3) (2003) 139-153
9. Kimia, B.B.: On the role of medial geometry in human vision. To appear, J. of Physiology, see www.lems.brown.edu/vision/publications/Kimia's_Publication/Journal/journals.htm.
10. LEMS, 99 silhouette database, www.lems.brown.edu/vision/software/shapes99.tar.gz.
11. Rom, H., Medioni, G.: Hierarchical Decomposition and Axial Shape Description. IEEE Trans. on Pattern Analysis and Machine Intelligence 15 (10) (1993) 973-981
12. Sebastian, T.B., Klein, P.N., Kimia, B.B.: Recognition of Shapes by Editing Shock Graphs. IEEE Trans. on Pattern Analysis and Machine Intelligence 26 (5) (2004) 550-571

13. Siddiqi, K.S., Shokoufandeh, A., Dickinson, S., S. Zucker, S.: Shock Graphs and Shape Matching. International Journal of Computer Vision 35 (1) (1999) 13-32
14. Torsello, A., Hancock, E.R.: A Skeletal Measure of 2D Shape Similarity. Computer Vision and Image Understanding 95 (2004) 1-29

Appendix – Finding the Global Reference Nodes

In order to compute the global distance between two shapes **A** and **B**, the global reference node pair given by $(\boldsymbol{p}_u^A, \boldsymbol{p}_v^B)$ must first be determined as summarized in Fig. 9.



START

Find merged segment SM_w^A in shape A with highest saliency

Find combination of secondary skeletal segments that can reconstitute merged segment SM_w^A with lowest total part distances (use only local part matching cost)

For each secondary skeleton in selected combination earlier, find its skeleton in shape B that gives the lowest part distance

For each secondary skeleton pair identified earlier, assume in turn, the start points of their skeletal segments (source nodes) are the global reference nodes. The pair of start points that gives the lowest total global distance error ξ_{gs} (see below) is the global reference node pair $(\boldsymbol{p}_u^A, \boldsymbol{p}_v^B)$.

END

Shape A
Most salient merged segment SM_w^A
Shape A
Lowest cost combination of secondary skeletal segments
Shape A
Shape B
p_u^A
p_v^B
Shape A
Shape B

Note: $\xi_{gs} = \sum_{i=1}^{NA} \min_{1\le j\le NB} [g_s(SS_i^A, SS_j^B)] + \sum_{j=1}^{NB} \min_{1\le i\le NA} [g_s(SS_i^A, SS_j^B)]$

NA and NB are the total number of secondary skeletons (SS_n) in shapes A and B respectively. And $g_s(\ )$ is the normalized difference in the global distances (highlighted in Fig. 3d and 3e) between the two segments. To normalized global distances, the spatial distance for each skeleton is divided by the square root of the shape area and angular distances are divided by π.

Fig. 9. Outline of procedure to determine the two global reference nodes for shapes **A** and **B**

Automatic Learning of Articulated Skeletons from 3D Marker Trajectories

Edilson de Aguiar, Christian Theobalt, and Hans-Peter Seidel

MPI Informatik, Saarbrücken, Germany
{edeaguia, theobalt, hpseidel}@mpi-inf.mpg.de

Abstract. We present a novel fully-automatic approach for estimating an articulated skeleton of a moving subject and its motion from body marker trajectories that have been measured with an optical motion capture system. Our method does not require a priori information about the shape and proportions of the tracked subject, can be applied to arbitrary motion sequences, and renders dedicated initialization poses unnecessary. To serve this purpose, our algorithm first identifies individual rigid bodies by means of a variant of spectral clustering. Thereafter, it determines joint positions at each time step of motion through numerical optimization, reconstructs the skeleton topology, and finally enforces fixed bone length constraints. Through experiments, we demonstrate the robustness and efficiency of our algorithm and show that it outperforms related methods from the literature in terms of accuracy and speed.

1 Introduction

Marker-based optical motion capture (MOCAP) systems reconstruct the motion of moving subjects by measuring the 3D trajectories of optical beacons attached to the body [1,2,3]. In order to biomechanically analyze the motion of a person or in order to map real world performances onto virtual characters, the captured marker-trajectories have to be transformed into the motion parameters of a kinematic skeleton model. Although commercial tools exist that assist the motion capture professionals in performing this transformation, the estimation of kinematic skeletons and their motion parameters is still a labor-intensive, error prone and often inflexible process. Many commercial systems require the tracked subject to strike a dedicated initialization pose (T-pose) prior to actual motion recording or need specific initialization movements. Moreover, due to measurement noise in the marker-trajectories and non-rigid deformations of the body surface commercial software often fails to enforce fixed bone length constraints.

Despite the relevance of the skeleton reconstruction and joint parameter computation problem, astonishingly few papers have been published that aim at solving it in an automatic, robust, flexible and more efficient way than standard software packages. We present a new algorithm to estimate a skeleton model and its motion parameters that does not require a specific initialization pose, that relies on a minimum of a priori knowledge about the kinematic structure, and that reconstructs a model with fixed bone lengths from arbitrary motion sequences. Our main contributions are:

- A new method to identify individual rigid bodies from 3D marker trajectories.
- A method to determine joint positions at each time step of video, and to extract the topology of a skeleton with fixed bone lengths that optimally captures the true body pose at each time step.

The remainder of this paper is structured as follows: Sect. 2 reviews the most relevant related work. Sect. 3 details our clustering procedure that is used to identify individual rigid bodies from the markers' 3D motion. Sect. 4 details how correct joint positions are found for each time step of video and how the skeleton topology is automatically inferred. An optimal skeleton with fixed bone lengths is computed thereafter by the method described in Sect. 5. We have tested our method on a large number of publicly available motion capture sequences and compared it to most related methods from the literature. Furthermore, we have validated our method on synthetic sequences which provides us with accurate ground truth information about the model's kinematic structure, Sect. 6. The paper concludes in Sect. 7.

2 Related Work

Nowadays, marker-based motion capture systems have developed into a standard tool within the technical repertoire of professionals in computer animation and biomechanical analysis. Unfortunately, generating a moving kinematic skeleton model from raw marker trajectories with commercial tools is often still a semi-automatic procedure [1,2,3]. Commercial software frequently requires the use of body models with predefined topology making it hard to capture subjects which are not stored in the model database. Furthermore, many tools fall short of providing skeletons with constant bone lengths and the IK-based joint parameter estimation often does not produce satisfactory results.

Most algorithms from the literature aim at solving one particular sub-problem in the overall motion capture pipeline. Biomechanics researchers have developed several methods to accurately locate the joint of a subject from the motion of bones or markers [4,5,6]. Other approaches are able to solve the skeleton reconstruction problem by taking into account a priori information [7,8]. O'Brien et al. [9] present a technique for determining the joint parameters of an articulated skeleton hierarchy from magnetic tracking data. In their work, both position and orientation information of the markers are available, which simplifies the skeleton reconstruction procedure. In contrast, we present an automatic method for jointly estimating an articulated skeleton and its motion from marker trajectories. Our method does not impose any constraints on the type of motion or type of subject being captured.

Most similar to our approach are the methods by Silaghi et al. [10] and Kirk et al. [11]. Silaghi et al. describe a semi-automatic approach to locally find skeleton structures. An optimal skeleton is then assembled by matching a template to the different skeletons found over time. Although skeleton models can be reconstructed reliably, their method requires a substantial amount of user interaction.

Kirk et al. present an automatic approach for determining the kinematic structure of a subject when only a small number of markers have been attached to it. Also with this method, articulation structures can be reconstructed. However, the complexity of the involved optimization problem makes it hard to apply their algorithm to long motion capture sequences with many body markers.

Our method builds on and improves ideas from the literature. It enables us to automatically identify rigid bodies, to automatically compute joint positions and skeleton topology, and to automatically enforce fixed bone length constraints. It does not require any a priori information about the tracked subject, is computationally efficient, and the quality of the estimated skeletons matches the quality of body models that have been generated with commercial tools.

3 Rigid Body Clustering

The input to our system is raw optical MOCAP data, i.e. 3D marker trajectories that can be acquired with all commercial optical MOCAP systems available today. Although the positional marker tracking accuracy achievable today is very high, some noise in the measurements is unavoidable. It is also a common problem that, due to self-occlusions on the body, some of the markers are temporarily invisible or even completely lost. In a pre-processing step, we eliminate from the trajectory data all the markers that are not visible in all the frames. In principle, these markers could still be used for improving the quality of the skeleton reconstruction in a post-processing step (e.g. by the method proposed in Kirk et al.[11]). However, our experiments have shown that a robust rigid body identification is possible even if only a few complete marker trajectories are at our disposition.

The first step in our processing pipeline is to cluster markers into groups, each of them representing one rigid body part. To serve this purpose, we capitalize on the fact that the distance between any two markers on the same body part remains constant (within a measurement tolerance) over time, while it varies if they lie on different parts. To robustly decide which markers lie on the same body part, we employ a spectral clustering algorithm that examines the standard deviations of the mutual marker distances over time. We make use of a fast variant of spectral clustering that has proven its robustness on many point segmentation problems [12]. In our implementation we define the entries of the affinity matrix $\mathcal{A}$ as follows:

$$\mathcal{A}_{i,j} = \exp(-\rho_{i,j}/(2 * \sigma^2)), \tag{1}$$

where $\rho_{i,j}$ is the standard deviation of the mutual distance between markers i and j over all frames, and $\sigma = 1/N^2 * \Sigma_f(dist^f_{i,j})$ is a scaling term controlling the spectral clustering convergence. N is the number of frames and $dist^f_{i,j}$ is the distance between markers i and j in frame f. Intuitively, the affinity matrix encodes the likelihood of each pair of markers to belong to the same body segment. Instead of grouping the markers directly based on the individual values $\mathcal{A}_{i,j}$, spectral clustering uses the top eigenvectors of matrices derived from

$\mathcal{A}$ to cluster the markers. This leads to a more robust and kinematically more meaningful segmentation than, for instance, the application of simple K-means clustering [13]. As an additional benefit, the optimal number of clusters N can be automatically calculated based on the datasets eigen-gap. Fig. 2(left) shows that our approach robustly identifies individual segments in the human body.

4 Estimation of Joint Positions and Skeleton Hierarchy

Given a list of body segments and their associated markers, we now estimate the positions of interconnecting joints at each time step of a motion sequence, and thereafter reconstruct the topology of the interconnecting bone skeleton.

The method to achieve the first goal makes use of a relatively straightforward observation. If we assume that two rigid bodies are connected via a single three-degree-of-freedom (DOF) ball joint then the distance between each marker on either of the adjacent bodies and the common joint has to remain constant over time. Taking measurement noise and subtle non-rigid body deformations into account, a good estimate for the correct joint position sequence is the sequence of points that minimizes the variance in joint-to-marker distance for all markers of the adjacent parts at all frames.

Kirk et al. [11] put this criterion into practice by computing the joint positions between two interconnected segments at all time steps via solving a large optimization problem. However, their approach is only feasible for sequences where the number of frames N and the number of markers M are small, since an energy minimization in $N * M$ variables for each pair of segments is necessary. In contrast, we have developed a faster scheme which efficiently finds optimal skeletons even with sequences that are several thousand frames long and which feature several hundred markers.

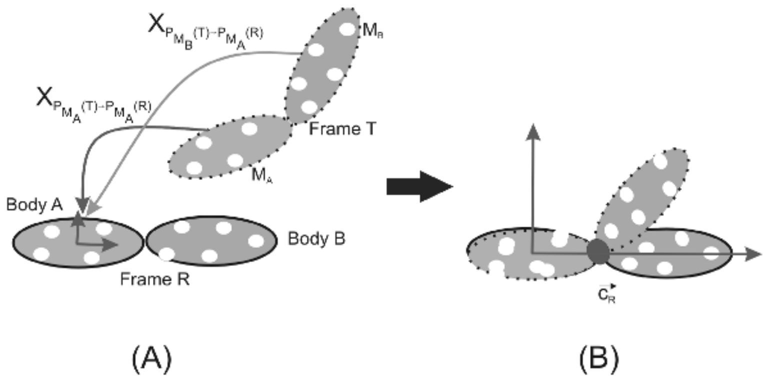


Fig. 1. Marker alignment: rigid body transformations are calculated (a) to align the position of markers for both segments in time step T with the markers of body segment A in the reference frame (b). After aligning the markers from all time steps the joint position $\boldsymbol{c_R}$ is found by minimizing (2).

Our scheme works as follows: Let A and B be two body segments, and let K be the set of the M body markers. Both body segments have associated sets of markers $M_A = \{a|a \in K\}$ and $M_B = \{b|b \in K\}$. At each time step f the markers in M_A and M_B have respective 3D positions $P_{M_A}(f) = \{\boldsymbol{p}(k,f)|k \in M_A\}$ and $P_{M_B}(f) = \{\boldsymbol{p}(k,f)|k \in M_B\}$. It is our goal to find the set $C = \{\boldsymbol{c_f}|f \in \{1,\dots,N\}\}$, i.e. the set containing the 3D position $\boldsymbol{c_f}$ of the interconnecting joint at each time step. To this end, we define a reference frame number R which can be any of the frames but usually is the first frame of the sequence. First, for each time step $t \in \{0,\dots,N\}$ we compute two rigid body transforms $X_{P_{M_A}(t)\to P_{M_A}(R)}$ and $X_{P_{M_B}(t)\to P_{M_A}(R)}$ that align the positions of the markers in both marker sets with the positions of the markers M_A at the reference time step [14], as shown in Fig. 1.

The positions of all markers at all time steps are now aligned with the marker positions at the reference time step. We are now able to solve for the joint location at the reference frame $\boldsymbol{c_R}$ by minimizing the following energy functional:

$$CF(\boldsymbol{c_R}) = 1/2 * \sum_{a \in M_A} (\sigma_a(\boldsymbol{c_R}) + \alpha * \overline{d}_a(\boldsymbol{c_R})) + 1/2 * \sum_{b \in M_B} (\sigma_b(\boldsymbol{c_R}) + \alpha * \overline{d}_b(\boldsymbol{c_R})) \quad (2)$$

where

$$\sigma_a(\boldsymbol{c_R}) = 1/N * \sum_{i=2}^{N} (\|\boldsymbol{c_R} - X_{P_{M_A}(i)\to P_{M_A}(R)} * \boldsymbol{p}(a,i)\| - \overline{d}_a(\boldsymbol{c_R}))^2 \quad (3)$$

and

$$\overline{d}_a(\boldsymbol{c_R}) = 1/N * \sum_{i=2}^{N} \|\boldsymbol{c_R} - X_{P_{M_A}(i)\to P_{M_A}(R)} * \boldsymbol{p}(a,i)\| \, . \quad (4)$$

The definitions of $\sigma_b(\boldsymbol{c_R})$ and $\overline{d}_b(\boldsymbol{c_R})$ correspond to (3) and (4). In (2), α is the coefficient that controls the influence of a distance penalty term. We employ the distance penalty term to prevent the algorithm from erroneously positioning the joint far away from either segment (e.g. infinitely away), where the variance $\sigma_a(\boldsymbol{c_R})$ and $\sigma_b(\boldsymbol{c_R})$ are minimal. Through experimental evaluation we have found that a value of $\alpha = 1/5$ leads to the best results. After finding $\boldsymbol{c_R}$, the joint position at all other frames can be computed by $\boldsymbol{c_f} = X^{-1}_{P_{M_A}(f)\to P_{M_A}(R)} * \boldsymbol{c_R}$.

Since we do not use a priori information about the topology of the subject, we perform the above procedure for each possible pair of body segments. Fortunately, the final values of the error term (2) enable us to automatically infer the skeleton topology and to discard invalid pairings of segments. To do so, we employ a graph-based method similar to the one presented in [9]. A skeleton graph is constructed in which each body part represents a node, and joints form the edges between them. Each edge is assigned a weight that corresponds to the value of (2) that we obtained for the pair of nodes (segments) that it connects. The topology of the skeleton can be determined by constructing the minimal spanning tree [15] of the skeleton graph.

Our method efficiently and robustly computes joint positions even for very long sequences with complex motion, as seen in Figs. 2 and 3.

5 Enforcing Constant Bone Lengths

Up to now, the lengths of the bones in the skeleton can vary from time step to time step. However, eventually one wants to express the motion parameters based on a single skeleton with constant dimensions. To serve this purpose, we have developed a simple and efficient way to enforce fixed bone length constraints in a separate processing step. We employ a least-squares fitting technique to appropriately adjust the joint positions that we have found by means of the approach described in Sect. 4.

Our algorithm follows the hierarchy of the estimated skeleton from the root down to the leaves (i.e hand/feet) and solves a least-squares problem for each pair of subsequent joints in the kinematic chain. By this means it is more efficient than related methods [10] that enforce fixed bone length constraints by solving a least-squares problem for the whole model at once.

Let us assume that $\boldsymbol{c_f}^i$ is the position of a joint i at frame f, and $\boldsymbol{c_f}^{i-1}$ is the 3D location of its parent joint at frame f. The optimal fixed length of the bone connecting joints $i-1$ and i, $l_{i-1,i}$, as well as the new joint positions of i, $\boldsymbol{oc_f}^i$, for all f can be found by minimizing the following cost function:

$$V(\boldsymbol{oc_1}^i, \ldots, \boldsymbol{oc_N}^i, l_{i-1,i}) = \sum_{f=0}^{N} \|\boldsymbol{c_f}^i - \boldsymbol{oc_f}^i\|^2 + (\|\boldsymbol{oc_f}^i - \boldsymbol{c_f}^{i-1}\| - l_{i-1,i})^2 \quad (5)$$

In (5) the first term is used to keep the new optimal joint positions as close as possible to the old positions, while the second term constrains the bone length to be the same in all frames. The dimension of the parameter space in (5) can be further reduced by expressing the new position of joint i in terms of the normalized direction vector $\boldsymbol{e}_{i-1,i}$ between $i-1$ and i. Replacing $\boldsymbol{oc_f}^i$ by $\boldsymbol{c_f}^{i-1} + \boldsymbol{e}_{i-1,i} * l_{i-1,i}$ in (5):

$$V(l_{i-1,i}) = \sum_{f=0}^{N} \|\boldsymbol{c_f}^i - (\boldsymbol{c_f}^{i-1} + \boldsymbol{e}_{i-1,i} * l_{i-1,i})\| \quad (6)$$

Eq. (6) is independently solved for each pair of subsequent joints in the hierarchy. The final result of our processing pipeline is a skeleton model of correct topology that, at each time step of motion, stands in a correct pose.

6 Results

We have tested our algorithm on a large number of optical motion capture sequences from the CMU motion capture database [16]. They were recorded with Vicon MX40 cameras. The motion sequences we used for testing comprise of 180-4000 frames and show, for example, simple gymnastic exercises, athletic performances and dancing sequences. After pre-processing of the raw data, on average around 110 non-interrupted marker trajectories were available for body model estimation. Fig. 2(left) shows the automatic segmentation result for a

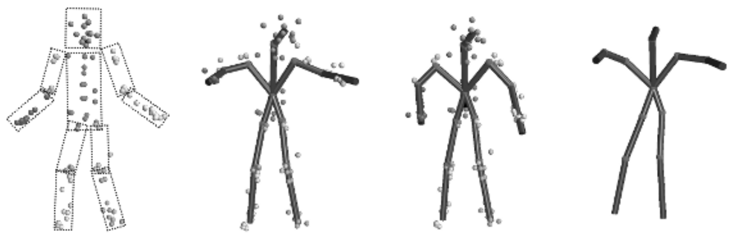


Fig. 2. Gymnastics sequence: Segmentation into body parts - boxes drawn for illustration purpose (left), two poses of the optimal skeleton shown together with the 3D marker positions (middle), and the fixed bone length skeleton at a different time step (right)

gymnastics sequence that was generated by our spectral clustering method described in Sect. 3. Individual segments of the human body have been correctly identified. Unfortunately, in the sequences that were at our disposition no significant foot or hand motion relative to the legs and arms respectively can be observed. In consequence, our algorithm cannot identify hand and feet as separate body segments. However, this is by no means a limitation of our method, but a general problem that is hard to solve for any learning-based approach.

Spectral clustering leads to much better segmentation results than, e.g., simple k-means clustering, since the clustering is far less deteriorated by noise in the data. While a purely distance based segmentation produces many kinematically meaningless rigid bodies, our variance-based scheme in conjunction with spectral clustering produces plausible body segmentations.

Fig. 2 shows different poses of the optimal kinematic skeleton reconstructed for some frames of a gymnastics sequence. One can see that both the topology of the bone skeleton and the positions of the joints have been faithfully estimated. The body models exhibit a high level of detail that is comparable to the complexity of skeletons usually used in animation and biomedical analysis. Fig. 3 shows further reconstruction results that we obtained by applying our algorithm to a dancing sequence.

Our method for estimating the joint locations and skeleton topology performs better than the method proposed by Kirk et al. [11] which is the most closely

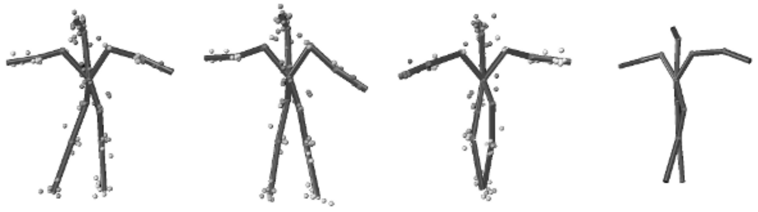


Fig. 3. Dancing sequence: The first three images show the markers and the estimated skeleton in three different poses. The image on the right shows the skeleton with constant bone length at another time step. Joint positions and skeleton topology have been faithfully reconstructed.

related approach from the literature (see also Sect. 4). As shown in Tab. 1, the runtimes of our method on complete motion capture sequences are orders of magnitude faster. We measured them on a Pentium IV with 3.0 GHz using the L-BFGS-B method to solve the minimization problems [17]. The comparison suggests that our approach is well-suited for processing long motion capture sequences as they are commonly recorded for most computer game and movie productions.

We have evaluated the accuracy of our system both visually and qualitatively. Unfortunately, we do not have ground truth measurements of the skeleton structures at our disposition. We thus compared our estimation results against the best possible reference data, which are the body models estimated with commercial software and that are provided by CMU together with their data. Fig. 4 shows visual comparisons for two different time steps. Our algorithm has reliably captured pose and dimension of the body. Please note that although our method reconstructed the topology of the root/spine area in a different way, the overall mobility is the same as in the reference model.

In order to get a qualitative error estimate, we have tested our method on synthetic data. Both test sequences (a walking robot and a jumping snowman) have been generated in 3D Studio Max by animating triangle meshes with handcrafted kinematic skeletons. In both test cases we use randomly selected vertices of the triangle meshes as markers for the reconstruction. Fig. 5(a) shows the robot in one pose and the respective skeleton reconstructed by our method. Kinematic structure and pose have been correctly identified. Fig. 5(b) shows that our approach correctly reconstructs model and pose in the case of the jumping snowman, too. In either test cases, the joint positions estimated by our method deviate on average by around 3% (relative to the model's height) from the true joint positions. This error is much lower than the position inaccuracy that we obtain with the method by Kirk et al. [11] which is in the range of 7%.

Our approach is subject to a few limitations. If the accuracy of marker trajectories is strongly deteriorated by noise or significant non-rigid body deformations (e.g. of the skin) are observed, joint positions may be improperly estimated. However, this is a general problem that commercial systems often fail to handle as well as the reference data provided by CMU suggest. Furthermore, it is impossible to distinguish two rigid body segments if at no time during a motion sequence a relative motion between them is observed. This is not a limitation specific to our approach but a general conceptual limitation of learning-based methods.

Table 1. Comparison of the runtime of our method to the runtime of the method proposed by Kirk et al. [11] on 4 different MOCAP sequences

Sequence	Number of Frames	Kirk et al. [11]	Our method (Sect. 4)
1	189	1649s	103s
2	307	2320s	175s
3	591	4515s	307s
4	1134	11247s	590s

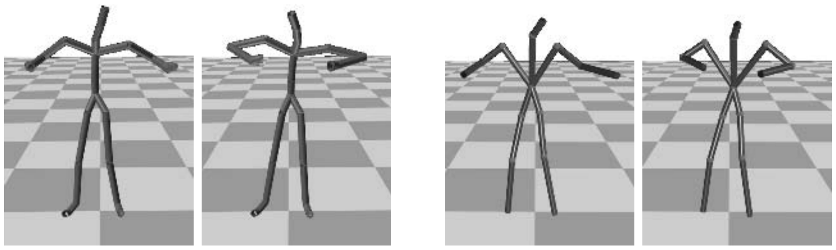


Fig. 4. Visual comparison of the skeletons that are provided with the MOCAP data (two images on left) and our learned skeleton in the same poses (two images on right)

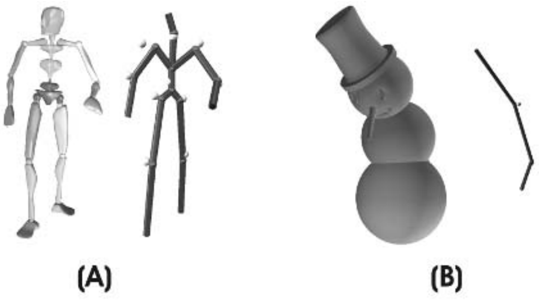


Fig. 5. Evaluation using synthetic data: (a) animated robot mesh (left) and reconstructed kinematic skeleton (right). Ground truth joints are shown as white spheres. (b) Animated snowman (left) and reconstructed skeleton with estimated joints (gray spheres) and ground truth joints (white spheres). The method is able to estimate the kinematic skeleton of general subjects accurately.

Despite these restrictions, our algorithm is an efficient and robust tool that can greatly simplify the motion capture pipeline. As shown, our method can be applied in the same way to motion data of arbitrary subjects including animals, generating accurate skeleton reconstructions.

7 Conclusion

We have presented a fully-automatic system for learning an articulated skeleton model with constant bone lengths and its poses from 3D marker trajectories. Our approach does with no a priori information about the kinematics of the captured individual and can be applied to arbitrary subjects including humans and animals. Through experimental evaluation we have shown that it performs better in terms of speed and accuracy than the most closely related methods from the literature. The learned models are comparable to the ones obtained with commercial software in terms of accuracy and detail. As future work, we plan to integrate our method with an automatic non-intrusive surface reconstruction approach in order to automatically learn complete virtual characters.

Acknowledgments. This work is supported by EC within FP6 under Grant 511568 with the acronym 3DTV.

References

1. Builder, V.B.: (http://www.vicon.com/products/bodybuilder.html)
2. Builder, M.A.S.: (http://www.motionanalysis.com/)
3. Motion, S.: (http://www.simi.com/)
4. Schwartz, M.H., Rozumalski, A.: A new method for estimating joint parameters from motion data. Journal of Biomechanics **38, Issue 1** (2005) 107–116
5. Gamage, S.S.H.U., Lasenby, J.: New least squares solutions for estimating the average centre of rotation and the axis of rotation. Journal of Biomechanics **35, Issue 1** (2002) 87–93
6. Spiegelman, J.J., Woo, S.L.: A rigid-body method for finding centers of rotation and angular displacements of planar joint motion. Journal of Biomechanics **20, Issue 7** (1987) 715–721
7. Cameron, J., Lasenby, J.: A real-time sequential algorithm for human joint localization. In Proc. SIGGRAPH'05, Posters(111) (2005)
8. Ringer, M., Lasenby, J.: A procedure for automatically estimating model parameters in optical motion capture. In: BMVC. (2002)
9. O'Brien, J.F., Bodenheimer, R., Brostow, G., Hodgins, J.K.: Automatic joint parameter estimation from magnetic motion capture data. In: Graphics Interface. (2000) 53 – 60
10. Silaghi, M.C., Plänkers, R., Boulic, R., Fua, P., Thalmann, D.: Local and global skeleton fitting techniques for optical motion capture. In: CAPTECH '98: Proceedings of the International Workshop on Modelling and Motion Capture Techniques for Virtual Environments, London, UK, Springer-Verlag (1998) 26–40
11. Kirk, A.G., O'Brien, J.F., Forsyth, D.A.: Skeletal parameter estimation from optical motion capture data. In: CVPR 2005. (2005) 782–788
12. Ng, A., Jordan, M., Weiss, Y.: On spectral clustering: Analysis and an algorithm (2001)
13. Duda, R.O., Hart, P.E.: Pattern Classification (2nd Edition). Wiley, New York - London - Sydney (2001)
14. Horn, B.: Closed-form solution of absolute orientation using unit quaternions. Journal of the Optical Society of America **4(4)** (1987) 629–642
15. Kruskal, J.B.: On the shortest spanning subtree of a graph and the traveling salesman problem. Proceedings of the American Mathematical Society **7** (1956) 48–50
16. Database, C.G.L.M.C.: (http://mocap.cs.cmu.edu/)
17. Byrd, R., Lu, P., Nocedal, J., Zhu, C.: A limited memory algorithm for bound constrained optimization. SIAM J. Sci. Comp. **16** (1995) 1190–1208

Real Time Hand Gesture Recognition Including Hand Segmentation and Tracking

Thomas Coogan, George Awad, Junwei Han, and Alistair Sutherland

Dublin City University, Dublin 9, Ireland
{tcoogan, gawad, jhan, alistair}@computing.dcu.ie

Abstract. In this paper we present a system that performs automatic gesture recognition. The system consists of two main components: (i) A unified technique for segmentation and tracking of face and hands using a skin detection algorithm along with handling occlusion between skin objects to keep track of the status of the occluded parts. This is realized by combining 3 useful features, namely, color, motion and position. (ii) A static and dynamic gesture recognition system. Static gesture recognition is achieved using a robust hand shape classification, based on PCA subspaces, that is invariant to scale along with small translation and rotation transformations. Combining hand shape classification with position information and using DHMMs allows us to accomplish dynamic gesture recognition.

1 Introduction

The primary goal of any automated gesture recognition system is to create an interface that is natural for humans to operate or communicate with a computerized device. Furthermore we aim to develop our system without using data gloves or colored gloves. Such a system could be used in, virtual reality, robot manipulation or gaming. In fact gesture recognition could be used to improve the intuitiveness of any Human-Computer Interaction (HCI). We routinely use hand gestures when communicating, describing and directing during our everyday activities. Incorporating gestures with HCI could be an extremely beneficial development.

In recent years various approaches to gesture recognition have been proposed, Gupta et al [1] presented a method of performing gesture recognition by tracking the sequence of contours of the hand using localized contour sequences. Chen et al [2] developed a dynamic gesture recognition system using Hidden Markov Models (HMMs). Patwardhan et al [3] recently introduced a system based on a predictive eigentracker to track the changing appearance of a moving hand. Kadir et al [4] describe a technique to recognize sign language gestures using a set of discrete features to describe position of the hands relative to each other, position of the hands relative to other body locations, movement of the hand, shape of the hand. While some of these approaches display impressive results, many exploit controlled environments and a compromise between vocabulary size and recognition rate.

To achieve accurate gesture recognition over a large vocabulary we need to extract information about the hand shape. Accomplishing this entails detecting the hands, segmenting them, differentiating them and classifying them. This requires using skin detection techniques and handling occlusion between skin objects to keep track of the status of the occluded parts. We present a unified system for segmentation and tracking of the face and hands in a gesture recognition using a single camera. Unlike much related work that uses color gloves [5], we detect skin by combining 3 useful features: color, motion and position. These features together, represent the skin color pixels that are more likely to be foreground pixels and are within a predicted position range. Also, unlike other work that avoid occlusions entirely by choice of camera angle, sign vocabulary, or by performing unnatural signs [6,7], we handle occlusion between any of the skin objects using a Kalman filter based algorithm.

Once the hand is segmented classification is required. In hand shape recognition, transformation invariance is key for successful recognition. We propose a system that is invariant to small scale, translation and shape variations. This is achieved by using a-priori knowledge to create a transformation subspace for each hand shape. Transformation subspaces are created by performing Principal Component Analysis (PCA) on images produced using computer animation. We introduce our method that enables us to train this appearance based method using computer animation images. Also presented is the incorporation of this hand shape classifier into a dynamic gesture recognition system. Position information is combined with hand shape information to construct a feature vector that is passed to a DHMM for dynamic gesture recognition.

The remainder of this paper is organized as follows: The method used to segment the face and hands is reported in section 2. The gesture recognition technique including hand shape recognition is described in section 3. In section 4 we detail some experiments and finally we offer some conclusions in section 5.

2 Segmentation and Tracking

In gesture recognition we need to segment and track three objects of interest: the face and the two hands. The skin segmentation module is responsible for segmentation of skin objects, similarly the object tracking module is responsible for matching the resulting skin blobs of the segmentation component to the previous frame blobs while keeping track of the occlusion status of the three objects. In the next sections we will explain the details of these two components.

2.1 Skin Segmentation

In order to robustly detect the skin objects, we combine three useful features: color, motion and position. Color cue is useful because the skin has a distinct color that helps to differentiate it from other colors. The motion cue is useful in discriminating foreground from background pixels. Finally, the predicted position of objects using Kalman filter helps to reduce the search space.

2.1.1 Color Information

In order to collect candidate skin pixels, we use two simple classifiers. First, a general skin model (color range) is applied on small search windows around the predicted positions of skin objects. As the fixed color range can miss some skin pixels, we propose another color distance metric dist(C_{skin}, X_{ij}) to take advantage of the prior knowledge of the last segmented object. This metric is the Euclidean distance between the average skin color C_{skin} in the previously segmented skin object and the current pixel X_{ij} in the search window at positions *i* and *j*. Finally, we normalize the values of the prior knowledge color metric P_{col}

2.1.2 Motion Information

Finding the movement information takes two steps. Firstly, motion detection, then next step, finding candidate foreground pixels. The first step examines the local gray-level changes between successive frames by frame differencing:

$$D_i(x, y) = |W_i(x, y) - W_{i-1}(x, y)| \tag{1}$$

Where W_i is the *i*th search window and D_i is the absolute difference image. We then normalize D_i to convert it to probability values. The second step assigns a probability value $P_m(x, y)$ for each pixel in the search window to represent how likely this pixel belongs to a skin object. This is done by looking backward to the last segmented skin object binary image in the previous frame search window OBJ_{i-1} and applying the following model on the pixels in D_i :

$$P_m(x, y) = \begin{cases} 1 - D_i(x, y) & \text{if } OBJ_{i-1}(x, y) \equiv 1 \\ D_i(x, y) & \text{otherwise} \end{cases} \tag{2}$$

In this way, small values (stationary pixels) in D_i that were previously segmented as object pixels will be assigned high probability values as they represent skin pixels that were not moved, and new background pixels with high D_i will be assigned small probability values. So simply, this model gives high probability values to candidate skin pixels and low values to candidate background values.

2.1.3 Position Information

To capture the dynamics of the skin objects, we assume that the movement is sufficiently small between successive frames. Accordingly, a Kalman filter model can be used to describe the x and y coordinate of the center of the skin objects with a state vector S_k that indicates the position and velocity. The model can be described as:

$$S_{k+1} = A_k S_k + G_k \tag{3}$$

$$Z_k = S_k + V_k \tag{4}$$

Where A_k is a constant velocity model, G_k , V_k represents the state and measurement noise respectively and Z_k is the observation. This model is used to keep track of the position of the skin objects and predict the new position in the next frame. Given that the search window surrounds the predicted center, we translate a binary mask of the object from the previous frame to be centered on the new predicted center. Then the distance transform is computed between all pixels in the search window and pixels of the mask. The inverse of this distance values assigns high values to pixels that are belonging to or near the mask and low values to far pixels. The distance values are then converted to probabilities P_{pos} by normalization.

2.1.4 Information Combination

After collecting the color, motion and position features, we combine them logically using an abstract fusion formula to obtain a binary decision image *F_i(x,y)*:

$$F_i(x,y) = \begin{cases} 1 & \text{if } (P_{col}(x,y) > \tau) \; OR \; ((P_g(x,y) == 1) \\ & \quad AND \; (P_m(x,y) > \upsilon) \; AND \; (P_{pos}(x,y) > \sigma)) \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Where P_{col}, P_m, and P_{pos} is the decision probability values of the color metric, motion, and position respectively. P_g is the output of the general skin model, and τ, υ, and σ are thresholds where σ is determined adaptively by the following formula:

$$\sigma = \frac{size\;((P_m(x,y) > \upsilon) \; AND \; (P_{pos}(x,y) \equiv 1))}{size\;(P_m(x,y) > \upsilon)} \tag{6}$$

The threshold σ determines the margin that we are searching into around the predicted object position. In Eq. (6) this is formulated by finding the overlapping between the predicted object position and the foreground pixels above certain threshold value. The other thresholds values are determined empirically.

2.2 Tracking and Occlusion Detection

2.2.1 Occlusion Detection

In this paper, we propose a Kalman filter based algorithm to detect occlusion between any of the face and the two hands. In general, the algorithm uses the Kalman filter model to track the four corner points of the bounding box around the face and two hands. This model can predict in the next frame the positions of these four corner points. Accordingly, we check to see if there is any overlap between any of the bounding boxes in the next frame. If there is an overlap, we raise an occlusion alarm corresponding to the two bounding boxes that will overlap.

If in the next frame, the number of detected skin objects is less than the current frame objects *and* an occlusion alarm was raised previously, we conclude that occlusion happened. On the other hand, if the number of detected skin objects decreases and no occlusion alarms were raised, then one or more skin objects have left the frame.

2.2.2 Tracking

The tracking process starts by first constructing search windows around each of the predicted positions of the tracked objects. When two or more objects are occluded, they are treated as one object and one search window is constructed around their position. Next, connected regions are labeled after removing noisy small regions. Using the number of detected skin objects and the occlusion alarms as discussed in section 2.2.1, we maintain a high-level understanding of the status of the current frame with respect to the occlusion status. For example, if we detected 1 object and occlusion alarm between the face and left hand is raised, then we conclude that the face and left hand are occluded and the right hand is hiding. This technique can be extended to handle all 7 situations of occlusion status: separate face and two hands, face and 2 hands occluded, separate hand with face and hand occluded, face and hiding hands, face and hands all occluded.

The final step in the tracking part is the blob matching where the previous frame blobs are matched against the new frame blobs using the knowledge of the high-level occlusions status we described above. The matching is done using the distance between the previous objects centers and the new objects centers.

3 Gesture Recognition

3.1 Hand Shape Recognition

To date many approaches have been proposed for hand shape recognition. These include: (i) Shamaie [8] introduced a PCA based approach. (ii) Just et al [9] introduced a hand shape system based on the Modified Census Transform MCT. (iii) A method to classify hand postures against complex cluttered background was proposed by Triesch & von der Malsburg [10] using elastic graph matching. (iv) Yuan et al [11] developed their system by determining a new Active Shape Model (ASM) kernel based on the shape contours. (v) Chen et al [2] present a method of classifying the static hand poses by using the Fourier Descriptor to characterize the spatial features of the hands boundary.

Many of these approaches for hand shape recognition display sub-optimal results due to the highly deformable nature of the hand. Once again a compromise is determined between small vocabulary and accurate recognition. Due to the complex temperament of the hand, any hand shape classifier needs to be able to cope with small rotations and translation transformations.

We propose a transformation subspace technique to combat these issues. Our proposed method creates the invariant subspaces from a sampled subset of all possible transformation images. These images are produced systematically using the commercially available POSER computer animation (CA) software and includes 3D hand transformation. Performing PCA on these images will generate a subspace that accurately represents the complex transformation hyper-plane.

Using this method of a-priori knowledge to construct the subspaces means we can eliminate the process of automatic subspace segmentation as proposed by [12]. It also allows us to dismiss the need for managing outliers or missing data in our subspaces [12]. This means we can create more accurate transformations subspaces to what was previously possible using the simple PCA method.

PCA provides M orthogonal eigenvectors {u1, … ,uM} of the covariance matrix, that correspond to the first M largest eigenvalues, in order to maintain a minimum energy of the dataset. In order to classify test images, a distance metric needs to be introduced. We project the test image into the subspace and find the perpendicular distance of the projected point to the eigenvectors representing the subspace.

We now have the backbone for hand shape recognition system. ISL contains 28 static fingerspelling gestures. The system is constructed as follows:

Training – Generating a transformation subspace for each hand shape.
Testing – Project the test image into each of the subspaces to find the subspace with the nearest perpendicular distance. This subspace will be representative of one particular hand shape.

3.2 Hand Image Pre-processing

If this technique is to be worthwhile we need to be able to accurately classify images of real hands when we train with CA images. In order to do these we need to neutralize the differences between CA hand images and images of a human hand. Such a system would inherently be multi-user as it would be trained and tested by different users. We have developed a detailed pre-processing step to counteract issues such as: skin color, illumination variation, hand size and distance from the camera.

3.2.1 Hand Scaling and Alignment

Once the hand has been identified and segmented it should be scaled and aligned to ensure the system can deal with users with different sized hands and users at differing distances from the camera. We exploit a simple but effective custom of scaling the hand objects so that they occupy a predetermined area in a 32*32 resized image. Alignment is accomplished by repositioning the hand object so that the center of the bounding box lies in the center of the image.

3.2.2 Skin Color and Illumination Variation

Removing skin color and color variance due to illumination is essential in an appearance based multi user hand shape recognition system. First the hand image is converted to grayscale, this reduces the space at which color can be represented. In order to color normalize each hand image in grayscale space we have incorporated a color histogram equalization approach into our system. Color histograms are graphs that depict the color distribution of pixels in an image. Histogram Equalization is the process of redistributing the color values in the image so that the image histogram takes a predetermined form.

We know from the hand-scaling step that all hand objects are resized to occupy the same area within an image. With this in mind, a common histogram can be defined that can represent all hand images. It contains a large spike that represents the background of the image; this is located at the beginning of the color scale because the background pixels are set to 0. Then a gaussian-shaped pulse exists towards the end of the color scale represents object pixels. This positioning is important to maximize the contrast of the normalized hand image.

3.2.3 Image Filtering

We have found that it is useful to apply a simple gaussian filter to an image before classification. This filter can help smooth out noise in an image. The hand image is convolved with a 9*9 gaussian kernel with a small standard deviation to ensure the filtering doesn't blur important information in the image.

3.3 Dynamic Gesture Recognition

Dynamic gesture recognition requires both temporal and spatial recognition of the hand movement and hand shape. We have devised a simple system using Hidden Markov Models (HMMs) to recognize dynamic gestures. A HMM is a tool for representing probability distributions over a sequence of observations. For our dynamic gesture recognition system the sequence of observations are feature vectors. These feature vectors consist of two elements, both of which are positive integers. The first denotes the group to which the static hand shape has been classified. It should be noted that 40 static hand shape groups are currently used to distinguish the gestures in this system. The second defines the position of the hand in the image and will be in the range 1-9. The position of the hand is classified by determining the section of the image that the center of the hand lies in. This center is calculated by finding the center of the hands bounding box.

The image is divided into 9 sections. These sections are created by diving the image vertically by drawing two lines either side of the head. The 1st of the horizontal lines, H1, is located directly under the head. The 2nd, H2, is placed M pixels below H1, where M is the length of the head object. Using this technique position can be calculated invariantly to the distance of the user from the camera. A gesture is then represented as a sequence of tuples, containing both shape and position information.

A DHMM is trained for each possible gesture using many different examples. A gesture is classified online, by manually identifying its start and stop point, then finding the DHMM with the highest probability for the feature vector of the test sequence.

4 Experiments

4.1 Static Hand Shape Classification

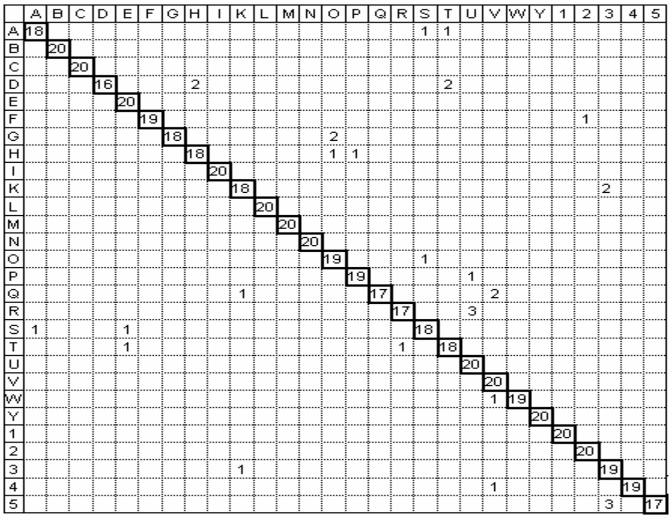
In order to classify real hand images we create a subspace for each hand shape as in section 3.1. A subspace for each hand shape is now created by performing PCA on 3969 images as depicted in equation 7.

$$\begin{aligned}&1\ \text{Origin image}^{1} \times 49\ \text{translations}^{2} \times 9\ \text{rotations in yaw direction}^{3}\\&\times 3\ \text{rotations in pitch direction}^{4} \times 3\ \text{rotations in roll direction}^{5}\\&= 3969\ \text{images}\ .\end{aligned} \tag{7}$$

[1] Origin hand image that can be defined as being the perfect orientation of the hand shape.
[2] Origin hand shapes translated in all directions using combinations of 2, 4 and 6 pixels.
[3] 9 rotations are used in the yaw direction as this is the direction that contains most significant deviation. These rotations are 3 degrees apart covering a total pitch of 24 degrees.
[4] 3 rotations in the roll direction, each at 10 degrees covering a total pitch of 20 degrees.
[5] 3 rotations in the roll direction, each at 10 degrees covering a total pitch of 20 degrees.

All test and training images are pre-processed using the techniques described in section 3.2. We developed a test set in order to test the amount of energy we need to retain in each of these subspaces. This test set contained 560 images, 20 occurrences of each of the 28 hand shapes been used. All these images were acquired from one trained user of the system over 4 separate sittings on 2 different days. In theses images the assumption has been made that the user is wearing long sleeves covering the arms. The first objective of our experiments was to identify the energy retention value that gives superior recognition. We have found that when 80% of the energy is retained the lowest error rate is achieved. One explanation for this is once we go over 80% the subspaces attempt to retain information that is local to that of the individual user, i.e. local characteristics of the computer animation images. It is important to find this balance between retaining as much information as possible without introducing noise in our subspaces. 80% energy retention entails keeping 12-16 of the most

Table 1. Confusion matrix for the 28 static handshapes



	A	B	C	D	E	F	G	H	I	K	L	M	N	O	P	Q	R	S	T	U	V	W	Y	1	2	3	4	5
A	18																	1	1									
B		20																										
C			20																									
D				16				2											2									
E					20																							
F						19																			1			
G							18							2														
H								18						1	1													
I									20																			
K										18																2		
L											20																	
M												20																
N													20															
O														19				1										
P															19					1								
Q										1						17					2							
R																	17			3								
S	1				1													18										
T					1												1		18									
U																				20								
V																					20							
W																					1	19						
Y																							20					
1																								20				
2																									20			
3										1																19		
4																					1						19	
5																										3		17

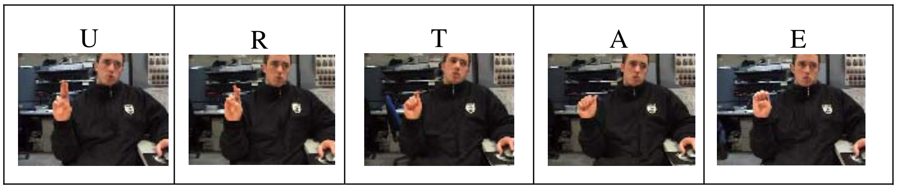


Fig. 1. Sample Static gestures

significant eigenvectors, depending on the hand shape. Having a low number of eigenvectors is also important to maintain efficiency.

Preserving 80% energy in the subspace we achieve 94.5% recognition accuracy for our test set. The performance accuracy of each individual static gesture can be observed in Table 1. This table exhibits the confusion matrix for the static gesture recognition vocabulary. Most confusion is caused where gestures are very similar. The two gestures that give highest confusion are U and R. These gestures only differ slightly as can be seen in Fig 1. When performing U, the index and middle finger lay parallel, while performing R the index and middle finger are crossed. These differences become particularly minute once the images are scaled to 32*32.

4.2 Dynamic Gesture Recognition

In order to test the accuracy of our dynamic gesture recognition system we have generated a vocabulary of 17 dynamic single-handed gestures. This lexicon was created to ensure gestures exist that differ only in either hand shape or hand position. 20 samples of each isolated gesture were recorded employing 2 different users, of different racial origins, over 4 different days. The videos are captured in an office environment with additional lighting to the front of the user. In our experiments the samples are divided into test and training sets by random sampling. The proportion of test and training data was then varied over the recognition experiments. A DHMM was trained for each gesture using the selected training data. We then tested the recognition accuracy using the remaining unseen data. Recognition accuracy was calculated by computing the average performance for different sampling of the training data. Table 2 displays this performance for each of the different number of data samples used in training. As expected the performance increases as the number of training samples increases. It is also interesting to note that reasonably high classification rates can be achieved using only one training sample for the DHMM.

This classification has been achieved on a standard PC using the matlab interpreter with non-optimized code in real time at 10 frames per second.

Table 2. Illustrates the performance for each of the different number of data samples used for training

No. Training samples	1	2	3	4	5	6	7	8	9	10
Average Performance	83.0	88.9	91.9	94.2	94.6	95.3	95.9	97.1	98.5	98.6

5 Conclusions

In this paper a detailed framework is presented for accurate real time gesture recognition. A unified approach for segmenting and tracking skin color objects has been described. Tracking helps to reduce the search space for segmentation while accurate segmentation helps to accurately enhance the tracking performance. Also accurate segmentation assists improving hand shape recognition using the subspace classifier described. A novel approach of training a subspace classifier using images generated from computer animation is also illustrated. Using image-processing techniques we have shown that accurate recognition is possible for human hands. Combining this hand shape information with the position information a gesture recognition system was generated. Successful classification was achieved for isolated gestures even with limited training. To improve performance over a larger lexicon we intend to introduce a more detailed position gauge, increase the bank of allowable hand shapes along with introducing new features such as hand motion.

Acknowledgements

This work is partly funded by the Irish Research Council for Science Engineering and Technology, (IRCSET).

References

1. Gupta, L., Ma S.: Gesture-Based Interaction and Communication: Automated Classification of Gesture Contours", IEEE Transactions on Systems, Man and Cybernetics – Part C: Applications and reviews, Vol 31, No 1, 2001

2. Chen, F.-S., Fu, C.-m., Huang, C.-L.: Hand Gesture Recognition Using a Real-time Tracking Method and Hidden Markov Models, Image and Vision Computing, Vol 21, pages 745-758, 2003.
3. Patwardhan, K. S. Dutta Roy, S.: Hand gesture modeling and recognition involving changing shapes and trajectories, using a Predictive EigenTracker, Pattern Recognition, (Article in Press), 2006.
4. Kadir,T., Bowden,R., Ong,E. J., Zisserman, A.:Minimal Training, Large Lexicon, Unconstrained Sign Language Recognition, In Proc BMVC'04, Vol 2, pp849-858, 2004.
5. Shamaie, A., Sutherland, A.: Hand Tracking in Bimanual Movements, Image and Vision Computing, 23, pp. 1131–1149, 2005.
6. Huang, C.-L., Jeng, S.-H.: A Model-Based Hand Gesture Recognition System, Machine Vision and Application, 12(5), pp. 243-258, 2001.
7. Terrillon, J.-C, Piplr, A., Niwa, Y., Yamamoto, K.: Robust Face Detection and Japanese Sign Language Hand Posture Recognition for Human-Computer Interaction in an Intelligent Room, In Proc. Int'l Conf. Vision Interface, pp. 369-376, 2002.
8. Shamaie, A.: Hand Tracking and Bimanual Movement Understanding, PHD Thesis, Dublin City University 2003.
9. Just, A., Rodriguez, Y., Marcel, S.: Hand Posture Classification and Recognition using the Modified Census Transform, in Proc. IEEE International Conference on Face and Gesture, pp 351-356, 2006.
10. Triesch, J., Von der Malsburg, C.: Classification of hand postures against complex backgrounds using elastic graph matching, Image and Vision Computing Vol 20, pp 937-943, 2002.
11. Yuan, Y., Barner, K.: An Active Shape Model Based Tactile Hand Shape Recognition with Support Vector Machine, in Proc 40th Annual Conf Information Sciences and systems, 2006.
12. Bishoff, H., Leonardis, A., Maver, J.: Multiple Eigenspaces, Pattern Recognition, Vol 35, pp 2613-2627, 2002.

Physically-Based Real-Time Diffraction Using Spherical Harmonics

Clifford Lindsay and Emmanuel Agu

Computer Science Department, Worcester Polytechnic Institute
100 Institute Road, Worcester, MA 01609-2280, USA
clindsay@wpi.edu, emmanuel@cs.wpi.edu

Abstract. Diffraction, interference, dispersive refraction and scattering are four wavelength-dependent mechanisms that produce iridescent colors. Wavelength-dependent functions need to be sampled at discrete wavelengths in the visible spectrum, which increases the computational intensity of rendering iridescence. Furthermore, diffraction requires careful sampling since its response function varies at a higher frequency variation with sharper peaks than interference or dispersive refraction. Consequently, rendering physically accurate diffraction has previously either been approximated using simplified color curves, or been limited to offline rendering techniques such as ray tracing. We propose a technique for real-time rendering of physically accurate diffraction on programmable hardware. Our technique adaptively samples the diffraction BRDF and precomputes it to Spherical Harmonic (SH) basis that preserves the peak intensity of the reflected light. While previous work on diffraction used low dynamic range lights, we preserve the full dynamic range of the incident illumination and the diffractive response over the entire hemisphere of incoming light directions. We defer conversion from a wavelength representation to a tone mapped RGB triplet until display.

1 Introduction

Real-time iridescence: Iridescent surfaces exhibit different colors (shimmer) as the view or incident light angles change. Diffraction (CD Roms), interference (oil slicks), dispersive refraction (glass prism) and scattering (rainbow) are four distinct optics mechanisms that cause iridescent colors. Iridescent colors are especially impressive when objects are moved interactively to fully demonstrate its angle-dependent nature. For instance, rotating a CD-ROM in real time, walking past an oil slick in a virtual world, or observing moving butterfly wings in real-time all give stunning visual effects. Consequently, the rendering of iridescent materials at interactive rates is attractive, but challenging.

Issues With Rendering Diffraction: Physically-accurate Bi-Directional Reflectance Functions (BRDFs) generate more photorealistic images since they capture the intricate light-surface interactions of the underlying phenomena. However, rendering physically-accurate wavelength-dependent iridescent BRDFs is complex because they need to be sampled over at many discrete wavelengths within the human visual range (380 - 700nm) [1]. Additionally, since the diffraction BRDF varies at a much higher frequency than interference, scattering and dispersive refraction, more samples are required to capture its full response. Due to these computational challenges, previous work has either rendered physically-based diffraction in offline renderers or used less computationally-intense approximate models for real-time rendering of the diffraction BRDF [13].

Real-time Physically-based Diffraction: In this paper, we demonstrate a technique to render physically-accurate diffraction at interactive rates. Our approach samples the high-frequency diffraction BRDF intelligently in order to preserve the peak value intensities for each maxima. We factorize both the diffraction BRDF and incident light using a low-order Spherical Harmonics (SH) basis such that evaluating the BRDF response function is reduced to a dot product at each sampled wavelength. Although Spherical Harmonics (SH) has traditionally been used for low frequency lighting, we are able to use SH for diffraction because our sampling approach aggressively reduces the required sampling frequency. Evaluating the diffraction surface responses at many wavelengths requires significantly more storage than traditional trichromatic (RGB) representations [15]. We reduce storage requirements on the GPU by using the Composite Model (CM) [14] for compactly storing diffraction reflections.

Our technique also permits the use of a full hemisphere of incident light as opposed to previous work [1,2,16] that was limited to point or area light sources. It is important to note that since diffractive response essentially separates incident white light into its wavelength (color) components, the use of a full hemisphere of light generates more realistic images.

The rest of the paper is as follows. Section 2 presents related work, section 3 gives some background that is necessary to understand our technique, section 4 describes our technique, section 5 presents our rendering results and section 6 concludes and describes future work.

2 Related Work

Diffraction: The majority of work on rendering physically accurate diffraction is designed for use in ray tracers and other offline renderers. Stroke [5] derived the geometric conditions for iridescence. Thorman [4] developed a computer graphics model for diffraction based on the grating equation. Thorman [4] applied Stroke's grating equation to derive an illumination model for diffraction and resolves specific issues with rendering iridescent colors. Nakamae et al [6] developed a technique for rendering natural objects including diffraction with high intensity light sources.

Agu [2], Stam [1] and Sun [16] have more recently developed shading models for rendering diffraction. Stam uses Fourier optics to develop an anisotropic reflection BRDF based on Kirchoff's approximation for light scattering. Agu developed a ray optics BRDF for rendering diffraction, which uniquely computes the peak intensity at the maxima for each viewing angle. Sun's model is based on wave optics and was applied specifically to rendering optical disks. All three diffraction models generate photorealistic images and can be rendered in real-time using today's graphics hardware ***provided the lighting environment is simple enough***. However, as more complex lighting is used, achievable frame rates drops quickly.

Spherical Harmonics: Recently, real-time rendering using Spherical Harmonics (SH) has focused on environment irradiance, BRDF factorization, and radiance transfer (such as inter-reflections, color bleeding, and caustics). Ramamoorthi and Hanrahan [10] developed an efficient irradiance environment map using Spherical harmonics that rendered diffuse surface reflection from an environment map. Westin, et al [12] developed a physically-based BRDF representation in terms of matrices of Spherical Harmonic coefficients. Ramamoorthi and Hanrahan improved on their previous model [9] to account for complex BRDFs using a Frequency Space analysis that represented BRDF as Spherical Harmonic coefficients. Sloan simultaneously developed a technique [8] for precomputing radiance transfer, which uses Spherical Harmonics for the lighting environment as well as for the diffuse BRDF, self-shadowing, and inter-reflections. Kautz *et al* [11] improved upon Sloan by generalizing the precomputation of radiance transfer for rendering arbitrary BRDFs using SH.

Lindsay and Agu [7] developed a technique that used a full spectrum wavelength color representation in conjunction with SH to develop a wavelength-dependent BRDF model for rendering iridescence. This work is most related to our work. However [7] renders interference not diffraction. The interference BRDF varies at a much slower rate than diffraction (extremely spiky with sharp peaks [14]) and requires less sampling. With the exception of [7], all SH work listed above use a trichromatic color representation for rendering reflections while our technique requires a wavelength representation.

3 Background

3.1 Physically-Based Diffraction BRDF

Stam [1], Agu [2] and Sun [16] have recently derived three alternative physically accurate BRDFs for diffraction. We felt that Agu's BRDF gave us adequate control of variables during SH factorization. Hence we utilized the Agu model for our technique. However, although we employ Agu's model, we emphasize that any of the three physically-based models can be used with our technique.

Agu's model [2] derives a closed form BRDF by applying Huygens-Fresnel principle for Fraunhofer Diffraction. Its main results are now summarized. Incident white light bounces off the CD-ROM grooves producing planar waves with uniform intensities. Typically, thousands of grooves occur per square inch on a CD-ROM with separations measuring about a wavelength of visible light. The Agu model sums the planar waves emanating from N grooves to produce a closed form expression. This closed form expression can easily be evaluated to determine output diffraction intensities for any pair of incident light and view angles.

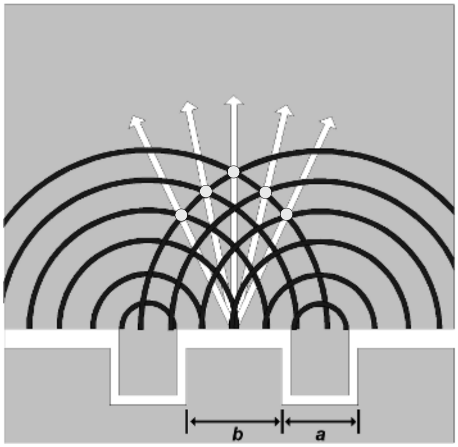


Fig. 1. Diagram of a measurement of similar locations in a single diffraction pit and groove. This diagram shows the modes (dots locating points where the light waves add together) and the measurement of $d = a{+}b$ from eq. 3.

Figure 1 shows one of the N diffraction grooves. Each groove has a width of dimension b followed by a gap of a distance a. Thus, $d = a + b$ is the distance between similar points on adjacent grooves. The incident light makes an angle θ_i with the normal vector of the CD-ROM and after diffracting off the CD-ROM, θ is the outgoing light angle. Equation 1 is the final expression for diffraction derived in [2]. For the complete derivation, we refer the reader to [2].

$$f(\Omega, \lambda) = Ambient + diffuse + \sum_{\Omega}\sum_{\lambda} I_{0\Omega\lambda}\frac{1}{N^2}\left(\frac{\sin\beta}{\beta}\right)^2\left(\frac{\sin^2 N\alpha}{\sin^2\alpha}\right) \quad (1)$$

$$\text{Where} \qquad \beta = \pi b\frac{(\sin\theta - sin\theta_i)}{\lambda} \qquad \text{and} \qquad \alpha = \pi(a+b)\frac{(\sin\theta - \sin\theta_i)}{\lambda} \quad (2)$$

$$\text{Since} \qquad \alpha = \frac{\pi d}{\lambda}(\sin\theta - \sin\theta_i) \qquad \text{then} \qquad d(\sin\theta - sin\theta_i) = m\lambda \quad (3)$$

where $m = 0, \pm1, \pm2...$ are diffraction modes. The ambient and diffuse terms of equation 1 are the same as for the Phong illumination model. Equation 3 is widely known as the *grating equation* and gives the locations of maxima. Figure 2(a) is a plot of equation 1.

The Nature of the Diffraction BRDF: We shall now highlight some key characteristics of the diffraction BRDF that makes it more challenging to render in real time than other iridescent phenomena. Since our general approach is to sample and factorize a physically accurate BRDF using SH, our main concern in rendering diffraction lies with the fast-varying nature of the diffraction BRDF function.

Figure 2(a) is a wavelength plot of the BRDF function in equation 1. In our work, a typical value of $N = 1500$ was used. The $\left(\frac{\sin^2 N\alpha}{\sin^2\alpha}\right)$ term causes the

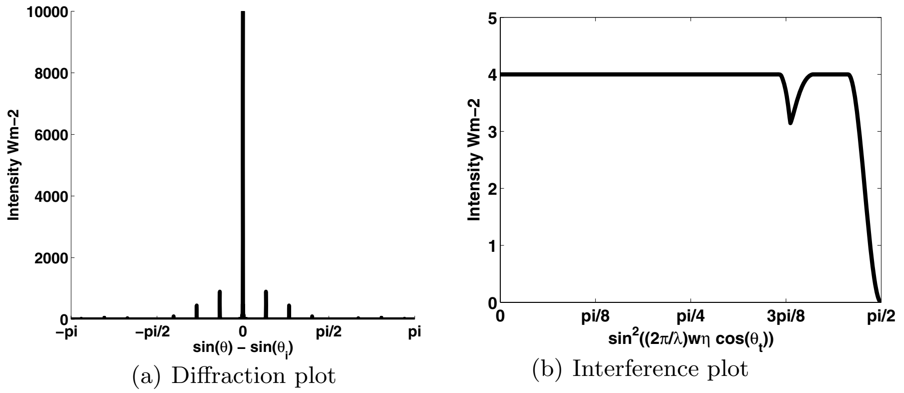


Fig. 2. These two diagrams show the relative intensity plots of the diffraction and interference. The lines showing delta functions for the diffraction plot have been exaggerated for demonstration and would normally be on the order nanometers or 1/100000th of a degree. From this diagram you can see that sampling of interference is less complex compared to diffraction.

function to vary very fast with extremely sharp narrow spikes. Sampling these peaks is important since they contain a large fraction of the BRDF response. These peaks are also contained within very narrow angular ranges such that angle changes on the order of 1/100000th of a degree (for a typical CD-ROM) can actually miss the location of the peak and result in erroneous intensity values. Capturing these peaks is non-trivial since for $N = 1500$, the diffraction curve can go from a minimum (0 response) to a maximum with a view angle change of 10^{-5} degrees.

However, not all wavelength-dependent phenomena exhibit these fast varying BRDFs. As a basis for comparison, we also plot the response of thin-film interference (soap bubbles) in fig 2(b). This slower varying interference function is not as challenging to sample as diffraction and was rendered using SH in [7]. In summary, accurate sampling, representation in terms of a wavelength basis and the efficient storage of samples are core issues with rendering diffraction.

Adaptive sampling the Agu Diffraction BRDF: Our goal is to determine the response intensity values for a given pair of incident light θ_i and view angles θ. To reduce the number of wasted samples, we invert and solve equation 3 to determine if there exists a diffraction peak for any mode m and wavelength λ. The modes, m can take integer values from -2 to +2 and λ can take values in the visible spectrum [380nm - 700nm]. If a valid combination of m and λ yields a peak in the visible spectrum, we then plug their values into equation 1 to determine the intensity of the peak. *Sampling only areas which have maxima, allows us to capture the narrow diffraction peaks using relatively few samples. Without using this sampling technique, the diffraction response would vary too fast for encoding using SH.*

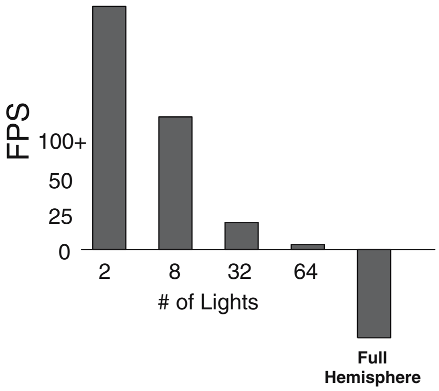


Fig. 3. Effect adding lights when directly evaluating diffraction BRDF. The horizontal axis shows the correlation between light complexity and frames per second (FPS). Evaluating a full hemisphere of light contribution results in negative FPS or requiring several seconds per frame to render.

Finally, it is important to compare rendering the physically-based diffraction BRDF directly versus our technique. A limitation of directly rendering the BRDF is that the frame rate is highly correlated with the number of lights used in the evaluation. Having a scene bound by the number of light is undesirable for real-time rendering and limits the light complexity. Figure 3 is a histogram showing the results of FPS versus light complexity. The left side of the axis shows a single light source and on the right a full hemisphere of light contributions. The frame rate for direct rendering drops dramatically with the number of light with a sampling rate of fifteen thousand stratified samples. As we mentioned previously, our technique does not suffer from the limitations of light complexity and runs at a constant frame rate due to our encoding the lighting in terms of SH coefficients.

3.2 Spherical Harmonics

Spherical Harmonics (SH) are a set of basis functions that can approximate a function in two dimensions defined over a sphere. In computer graphics and image synthesis, a 2D signal can be approximated in spherical coordinates. In our case, we approximate smooth distant lighting and diffraction BRDFs as a set of low-frequency weights for the real values SH basis. In general diffraction is considered a high-frequency reflection. As we describe in section 3.1 our sampling technique allows for representing the diffraction as a medium-frequency signal. This affords us the opportunity to approximate the diffraction with SH. SH produces a finite or band-limited approximation of a function by truncating the infinite sum of an integrable function to a countable set of coefficients. Equation 4 is a common form for real-valued Spherical Harmonics used in computer graphics. For a more complete description of SH, we refer the reader to [10, 11].

$$y_l^m(\theta,\phi) = \begin{cases} \sqrt{2}K_l^m \cos(m\phi)P_l^m(\cos\theta), & \text{when } m > 0, \\ \sqrt{2}K_l^m \sin(-m\phi)P_l^{-m}(\cos\theta), & \text{when } m < 0, \\ K_l^0 P_l^0(\cos\theta), & \text{when } m = 0 \end{cases} \quad (4)$$

3.3 Composite Model

An accurate, efficient and compact color model is needed to represent color for real-time diffraction rendering such as the Composite Model (CM) [14]. CM simply states that any wavelength-dependent function $S(\lambda)$ (e.g. diffraction) can be decomposed into a smooth and spiky. Since our diffraction function generates a series of spikes, we found it useful to store them using the spiky part of the CM. We do not use the smooth part of the CM. Each delta function is stored as its intensity value along with their corresponding wavelength location in the visible spectrum, resulting in a compact representation.

The behavior of reflections caused by diffraction result in intensity spikes in a very narrow region on the visual spectrum [3]. Minima and non-maximal points contribute almost nothing to the final output. The justification for employing The CM for diffraction color representation is that if we ignore the smooth decomposition for the spectral function because of its insignificance to the overall reflection, we can represent diffraction as a series of delta functions (about 4 per wavelength [3]). Each maximum of our diffraction calculation will result in a single delta function or spike. Storing the necessary information to represent the delta function consists of an intensity value and a single location on the visible spectrum. The storage requirement for this spectral representation is constant and therefore extremely compact.

4 Our Technique for Real-Time Diffraction

4.1 Overview

In this section, we describe our technique for real-time rendering of physically based diffraction using SH. Our technique maintains an accurate representation of physically based diffraction without sacrificing efficiency. To accomplish this, we separate the rendering of diffraction into three phases: a precomputation, rotation and upload, and rendering. A visual depiction of our pipeline is outlined in Figure 4.

Precomputation begins with an offline process that samples and projects both the lighting input and diffraction simulation. After projection, the resulting SH coefficients are stored on the CPU, which is signified by the arrows spanning the precomputation stage and CPU stage in Figure 4. The lighting coefficients are then rotated to the tangent frame on the CPU, and uploaded to the GPU. The texture map of SH coefficients is also uploaded to texture memory on GPU. In a vertex shader, the view vector is transformed to tangent space and used as texture coordinates to lookup the SH texture map. The dot product of the texture map diffraction coefficients and the rotated lighting coefficients result in the final wavelength, which is converted to an RGB triplet and uploaded to the framebuffer.

In the following sections, we describe in more detail our precomputation and rendering steps for rendering diffraction. As mentioned in earlier sections, for clarity of exposition, we focus on one specific diffraction BRDF, but our method is general enough to be applied to other diffraction BRDFs.

4.2 BRDF and Lighting Precomputation

The primary goal of the precomputation stage is to generate a hemisphere of light around a point on the diffraction surface and enumerate the diffraction responses for all valid incoming light and outgoing view directions. Separating the

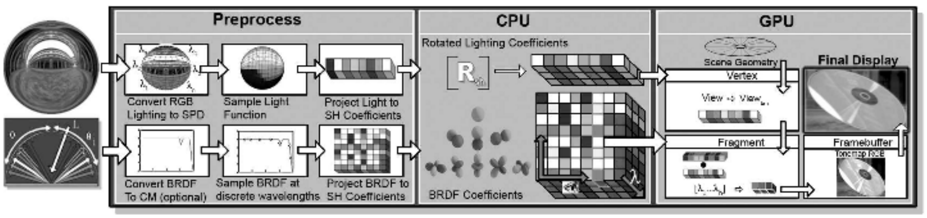

Preprocess
CPU
GPU
Convert RGB Lighting to SPD
Sample Light Function
Project Light to SH Coefficients
Convert BRDF To CM (optional)
Sample BRDF at discrete wavelengths
Project BRDF to SH Coefficients
Rotated Lighting Coefficients
BRDF Coefficients
Scene Geometry
Vertex
Fragment
Final Display
Framebuffer
Tonemap RGB

Fig. 4. Block diagram outlining the complete rendering pipeline for our technique. The pipeline is divided into three distinct stages: preprocessing, rotation and transfer, and rendering. The first stage is responsible for sampling and projection of lighting and BRDF to an SH basis. The second stage rotates the lighting coefficients and upload both lighting and BRDF coefficients to the rendering context. The third stage is responsible for the final output.

BRDF from the lighting until light integral calculation at run-time allows us to dynamically change lighting values without having to run the computationally expensive BRDF simulation. By varying the viewing direction, the SH approximation of the diffraction BRDF can be tabulated by view direction in a texture. A small set of SH coefficients are calculated per view. In many cases, at most 25 SH coefficients per view direction are required for most arbitrary BRDFs [11]. The size of the texture depends on parameterization of the view vector, spherical samples on s, and the desired accuracy of approximation (coefficients/view). Equation 5 outlines the projection of an arbitrary BRDF and the resultant set of coefficients c_i where $f(s)$ can be an arbitrary diffraction BRDF and $y_l^m(s)$ is a general form of the SH basis defined in section 3.2.

$$c_i = \tilde{f}(s) = \int_s f(s) y_l^m(s) ds \tag{5}$$

Precomputing the lighting coefficients from different light sources *L(s)* has been outlined in several previous papers [8, 11]. We improve on previous techniques for precomputing lighting by using the CM for representing the spectral power distribution instead of RGB triplets. The precomputed vector of coefficients l_i representing the lighting environment can also be used for the non-diffraction part of the BRDF. The precomputation is done by projecting a captured or simulated lighting environment into the SH basis (see equation 6). In our case, we project an image-based lighting environment that was generated from a series of High Dynamic Range photos (light probe, courtesy of Paul Debevec).

$$l_i = \tilde{L}(s) = \int_s L(s) y_l^m(s) ds \tag{6}$$

4.3 Rendering

The following section presents a detailed description of the steps to utilize the precomputed lighting and BRDF coefficients generated in section 4.2, for real-time rendering. The outlined rendering steps can be performed on a CPU or GPU, and may be used in both hardware and software renderers including shader-capable GPUs. One advantage of our technique is that the rendering phase uses only standard graphics operations, such as texture mapping, transformations, and simple mathematical operations. In fact, the presented technique can be seamlessly integrated with other interactive rendering techniques to improve overall photorealism. Figure 5 is our algorithm for rendering diffraction.

In step 1, for efficiency, the lighting is transformed once per vertex instead of per fragment. For a thorough explanation of rotating Spherical Harmonic coefficients refer to the appendix in [11]. Steps 2,3 are standard shader programming techniques for transforming and rotating vectors. Step 3 refers to a local tangent frame in which the view vector needs to be transformed and rotated. The surface normal and two orthogonal vectors in the texture direction define the tangent frame. For efficiency, the tangent space basis is pre-calculated offline and uploaded to the vertex shader at runtime. Once the view vector has been transformed/rotated, it is used as texture coordinates to look up SH diffraction coefficients.

The texture map of SH coefficients comprise the precomputation of the diffraction response to arbitrary distant lighting. A listing of view-dependent diffraction

Algorithm. Diffraction Rendering Steps

```
1. Rotate lighting coefficients, $l_i$, into appropriate frame on the host CPU.
2. Upload the lighting coefficients to the shader.
3. Transform and rotate the view vector to the tangent frame.
4. Look up the diffraction coefficients $c_i$ in a texture,
5. Scale and bias diffraction coefficients to the ($0 \Rightarrow 1$ to $-1 \Rightarrow 1$) range.
6. Apply the dot product of the coefficients.
7. Convert from wavelength to RGB, then upload to the framebuffer.
8. Tone map the framebuffer from high dynamic range to a displayable range.
```

Fig. 5. Step-by-Step algorithm for rendering diffraction

responses encoded into SH coefficients is stored in an $N \times V$ texture , where N is the number of SH coefficients used (we used 25), and V represents all potential view directions. Each texel in the texture map represents a portion of the Spectral Power Distribution (SPD) that will eventually be converted to RGB values. The SPD encoding method dictates the number of textures required. For example, a CIE color representation may use only one texture for the X, Y, Z values. For Agu's diffraction model, the response in a given view direction is essentially a delta function at a certain wavelength. The resulting texture map utilizes only two values (wavelength and intensity) per texel.

The dot product of the SH coefficients approximates the lighting integral. Evaluating this integral on the GPU when available, further enhances rendering speed. The computed Spherical Harmonic approximation of lighting and diffraction have unbounded dynamic range. Therefore, the reflection calculations result in color values outside the displayable range and need to be tone-mapped to the range of the target viewing system. Figure 5 is the necessary steps for the rendering algorithm on programmable GPUs.

5 Results

We achieved real-time frame rates with our technique without sacrificing image quality. As mentioned earlier, our technique's frame rate does not reduce as more lights are added to the scene, unlike direct evaluation of the diffraction BRDF. Figure 5 shows our final results, which were rendered on an Nvidia 6600 GT graphics card with 128 Megs of RAM on a Pentium 4 2.8 GHz Personal Computer with 1GB RAM. For our CD-ROM model that had 25 thousand vertices, we were able to maintain a constant frame rate of 65 Frames Per Second (FPS). With several additional effects such as environment mapping, blur, exposure control, and vignette, our scene still ran at over 60 FPS (real-time frame rate).

Limitations of our work: Precomputation time is a factor to consider when employing our technique. Additional computation time is required for spectral conversion making our diffraction BRDF significantly longer to precompute than other BRDFs [9, 8, 11] that are not wavelength-dependent. We see this as an area for considerable improvement as other wavelength-based phenomena may require more time for precomputation.

Our use of SH made spherical mapping an obvious choice for parameterization during precomputation but this resulted in too few samples at the equators. Storing the precomputed diffraction response using parabolic mapping [17] would produce more evenly distributed samples and result in fewer sampling errors. Additionally, highlights in the lighting directions exhibit an exponential fall off as the diffractive response moves away from the light direction. The high frequency nature of the fall off requires higher sampling than SH can achieve with low order coefficients ($\leq$ 25) which can lead to approximation errors. In future, different sampling techniques, factorization, or high order approximation with compression (for example PCA) may be able to capture the fall off exhibited by CDs with lower error.

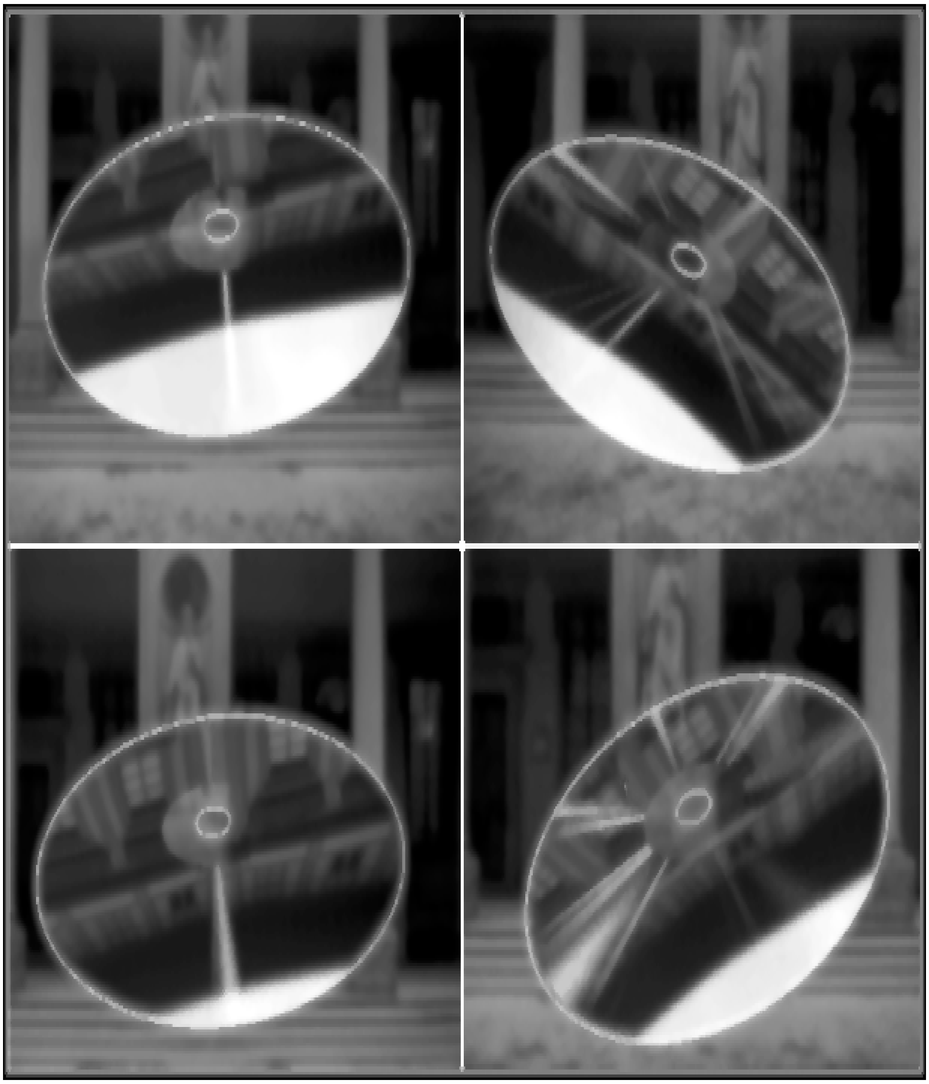


Fig. 6. Our final images: The top row of images are achieved by directly evaluating our physically-based diffraction BRDF with a single light rendering on the left and multi-light rendering on the right. The bottom set of images have been rendered with our technique using a single light on the left and a multiple lights on the right. Each of the SH approximated images (bottom) renderings run at the same frame rate even though there are 5 more lights in the multi-light version. Since SH Lighting is not affected by the number of light sources, a single light rendering runs at the same speed as full hemisphere lighting. The scene frame rate reduces as more lights are added, if the diffraction BRDF is evaluated directly.

6 Conclusion and Future Work

We have demonstrated a real-time technique for rendering physically based diffraction using Spherical Harmonics for compact and efficient representation. Our technique preserves the accuracy of the diffraction for arbitrary surfaces, including unique intensities for each response maximum, without suffering from a slower render time as the number of light sources increase. We believe that this technique can expand the range of complex materials currently rendered in real-time and enrich experience in many interactive applications such as virtual reality and video games.

In the future, we think that a normal progression of our technique would be to expand it to other wavelength dependent phenomena beyond iridescence and diffraction type reflection. Some techniques that may benefit from our technique include dispersion, fluorescence, and wavelength dependent multiple scattering.

References

1. Stam, J., *Diffraction Shaders.* in Proc. ACM SIGGRAPH 1999.
2. Agu, E., *Diffraction Shading Models for Iridescent Surfaces.* in Proc. IASTED VIIP 2002
3. Agu, E., *Diffraction shading models in computer graphics.* (PhD Dissertation, U. of Mass-Amherst).
4. Thorman, S., *Diffraction based models for iridescent colors in computer generated imagery.* (PhD Dissertation, U. of Mass-Amherst).
5. Stroke, G. W., *Diffraction Gratings, pp 426-754, Handbook Der Physik (Encyclopedia of Physics).* Springer-Verlag, 1967.
6. Nakamae, E., Kaneda, K., Okamoto, T. and Nishita, T., *A lighting model aiming at drive simulators.* in Proc. ACM SIGGRAPH '90, pg. 395-404
7. Lindsay, C., Agu, E., *Wavelength dependent Rendering Using Spherical Harmonics.* in Proc. Eurographics 2005
8. Sloan, P., Kautz, J., Snyder, J., *Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments*, ACM Transactions on Graphics, pg 527-536, 2002.
9. Ramamoorhthi, R., Hanrahan, P., *Frequency Space Environment Map Rendering.* in Proc. ACMSIGGRAPH 2002, pg 517-526.
10. Ramamoorthi, R., Hanrahan, P., *An Efficient Representation for Irradiance Environment Maps.* in Proc. ACM SIGGRAPH 2001, pg 497-500.
11. Kautz, J., Sloan, P., Snyder, J., *Fast Arbitrary BRDF Shading for Low-Frequency Lighting Using Spherical Harmonics.* in Proc. 13th Eurographics workshop on Rendering, pg 291-296, 2002.
12. Westin S., Arvo J., Torrance K., *Predicting Reflectance Functions from Complex Surfaces.* in Proc. ACM SIGGRAPH 1992, pg 255-264.
13. Stam, J., *Simulating Diffraction*, Chapter 8, in GPU Gems, Addison Wesley, 2004.
14. Sun Y., Drew, M., S., AND Fracchia F.,D., *Representing Spectral Functions by a Composite Model of Smooth and Spiky Components for Efficient Full-Spectrum Photorealism.* in Proc. IEEE Workshop on Photometric Modeling for Computer Vision and Graphics '99, pp. 4-11.

15. Sun Y., *A Spectrum-Based Framework for Realistic Image Synthesis*. Phd dissertation, Simon Fraser University, 2000
16. Sun Y., Fracchia F. D., Drew M. S., and Calvert T.W., "Rendering Iridescent Colors of Optical Disks," Simon Fraser Univ., Tech. Report SFU CMPT TR 1999-08, 1999.
17. Heidrich W., *View-Independent Environment Maps*. in Proc. Eurographics/SIGG RAPH Workshop on Graphics Hardware '98'.

3D Segmentation of Mammospheres for Localization Studies*

Ju Han[1], Hang Chang[1,2], Qing Yang[2], Mary Helen Barcellos-Hoff[1], and Bahram Parvin[1]

[1] Lawrence Berkeley National Laboratory, Berkeley, CA 94720
[2] Institute of Automation, Chinese Academy of Sciences, Beijing, China

Abstract. Three dimensional cell culture assays have emerged as the basis of an improved model system for evaluating therapeutic agents, molecular probes, and exogenous stimuli. However, there is a gap in robust computational techniques for segmentation of image data that are collected through confocal or deconvolution microscopy. The main issue is the volume of data, overlapping subcellular compartments, and variation in scale and size of subcompartments of interest. A *geometric* technique has been developed to bound the solution of the problem by first localizing centers of mass for each cell and then partitioning clump of cells along minimal intersecting surfaces. An approximate solution to the center of mass is realized through iterative spatial voting, which is tolerant to variation in shape morphologies and overlapping compartments and is shown to have an excellent noise immunity. These centers of mass are then used to partition a clump of cells along minimal intersecting surfaces that are estimated by Radon transform. Examples on real data and performance of the system over a large population of data are evaluated. Although proposed strategies have been developed and tested on data collected through fluorescence microscopy, they are applicable to other problems in low level vision and medical imaging.

1 Introduction

Current models of high throughput and high content screening are based on two dimensional cell culture assays that are grown either on plastic or glass. Although such a model system may be appropriate as an initial step toward discovery or certain aspect of biological studies, the knowledge may not readily extensible to *in vivo* models. On the other hand, animal studies are expensive and time consuming and as a result cannot scale for high throughput studies that is necessary to build a space-time continuum of responses in the presence of biological heterogeneity. An intermediate step is three-dimensional cell culture

* The Research was supported by National Aeronautics and Space Administration Grant no. T6275W, NASA Specialized Center for Research in Radiation Health Effects, the low dose radiation research program and medical imaging program at the Office of Biological Effects Research U.S. Department of Energy, Grant No. DE-FG03-01ER63240. PubID is LBNL-61402.

model systems that has been demonstrated to have some of functionalities of the *in vivo* models [11]. However, such a model system introduces significant computational challenges: (i) imaging is in 3D and not in projection space, (ii) subcellular compartments often overlap and delineation is made difficult, and (iii) variations in subcellular scale imposes a more complex segmentation problem at the object level. In this paper, we present a series of unique geometric steps for segmentation of 3D cell culture models, also known as acini, that enables subsequent localization studies and protein-protein interactions.

Research in the analysis of subcellular structures spans texture-based features for classifying patterns of protein expression [5], and geometric methods [12,8,9] and surface evolution methods [4,6] for delineation of nuclear compartments. Segmentation provides context for quantifying protein localization in fixed samples with an antibody or a nucleic acid based probe in living cell studies [8]. To our knowledge, previous techniques are not applicable for automated segmentation mammospheres. Proposed solution is to bound an inherently ill-posed problem through geometric constraints. A key observation is that nuclear regions are often convex and form a positive curvature maxima when they overlap each other. This feature was used earlier in 2D segmentation of nuclear regions [9]. However, evaluating 3D convexity and estimating 3D surface curvature is hindered with a significant computational complexities. Convexity can be viewed as a salient feature, which is an important perceptual cue for localization and segmentation of subcellular regions. In biological image understanding, saliency can be driven by continuity, symmetry, convexity, or closure. Among these, it is well known that symmetry is a pre-attentive process [1] that improves recognition, provides an efficient mechanism for scene representation, and aids in reconstruction and description. Radial (spherical in 3D) symmetry is a special class of symmetry, which persists in nature at multiple scales. Robust and efficient detection of inexact radial symmetries facilitates the semantic representation, interpretation, and prioritization for higher level processes. Yet, the notion of radial symmetry is used in a weak sense since the basic geometry can deviate in aspect ratio and convexity. Localization of approximate centroid of each nucleus in a three dimensional cell culture assay enables partitioning a mammosphere along the planes that generate minimum surface cross sections and are possibly aligned with points of maximum curvature along the surface.

The novelty of the proposed method is in specific geometric steps designed to bound the solution through seeding and subsequent partitioning. The basis for seeding (e.g., estimating centers of mass) is through geometric voting and perceptual grouping, and is implemented through the refinement of specifically tuned voting kernels [13]. Localization of centers of mass for each nucleus provides a bound on overall stable segmentation. Partitioning of adjacent nuclei is performed by finding planes that best separate adjacent cells, and the actual methodology is based on the Radon transform. In comparison, scale space representation (e.g., Gabor filters and scale-invariant feature transform) is less than adequate since they don't incorporate global geometric constraints. For example, our experience indicates that such methods can infer one seed for two overlapping nuclei. This is often due to the fact that (i) nuclear size varies widely, and (ii) intensity distribution within the nuclear region is non-uniform.

The organization of this paper is as follows. Section 2 provides a brief review of the previous research. Section 3 describes the basic idea and detailed implementation of approach. Section 4 demonstrates the experimental results. Section 5 concludes the paper.

2 Previous Research

Current state of art for delineation of nuclear regions from 3D multicellular systems and mammospheres leverage intensity information with limited amount of inherent geometry. Some of these methods are interactive and serve as a computed aided tool to increase operational throughput. In [7], background and nuclear regions are automatically delineated using a thresholding mechanism, Hough transform and automatic focusing are applied to estimate the size of the nuclei, the user labels each object as a single nucleus or a cluster of nuclei, and the process ends with watershed method to partition potential clumps of cells. In [10], limitations in [7] were identified in several computational steps: (1) initial thresholding, (2) noise, and (3) low gradient in some of the nuclear regions. These limitations were then addressed using level set methods for improved performance. In [12], nuclear regions were modeled as elliptic features and fragmented features were grouped together to form a convex hull. The method produces a segmentation that is not very accurate along the surfaces with potential fragmentation of nuclear regions. Recently, in [2] mammosphere slices were segmented in 2D and then segmented 2D slices were merged together. However, combination of assay and high resolution imaging produced a morphological nuclear signature that tend to be more separable in 2D (e.g., little overlap) while maintaining similar scale in nuclear size. One novelty of this system is that the analysis does not require isotropic representation of the data volume.

In contrast to previous approach, proposed method uses high level geometric features to delineate a multicellular system. Geometrically, nuclear regions are almost convex; however, scale (e.g., size) is heterogeneous. Furthermore, when two nuclear regions overlap, they form folds corresponding to positive curvature maxima. Partitioning adjacent nuclear regions along points of curvature maxima or a variation of that is the final step of the process.

3 Approach

Proposed steps in delineation of nuclei in a mammosphere system are shown in Figure 1. Starting from the interpolated 3D image, the solution first bounds the problem by computing seeds that estimate the centroid of each nucleus through iterative radial voting in 3D. Simultaneously, the colony is thresholded in 3D, which produces an erroneous segmentation of the clump of nearby cells by merging them. Each clump is subsequently labeled for further analysis, and any connected volume with more than two seeds is subject for further analysis.

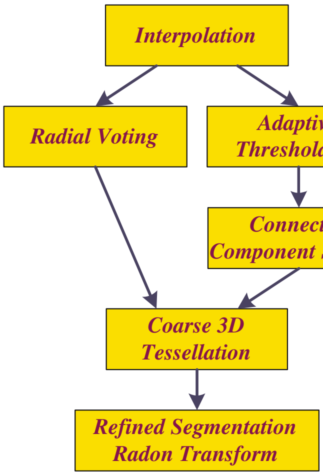


Fig. 1. Detailed representation of methodology for segmentation of a 3D mammosphere

Partitioning is performed by finding planes that best separate adjacent nuclei, and the actual methodology is based on the Radon transform. However, Radon transfer precedes by a coarse segmentation from adjacent seed locations. Ideally, such a coarse segmentation should be realized through voronoi tessellation, which is compute intensive in 3D. A simpler approximation to voronoi tessellation is implemented to provide a rough segmentation of nuclear regions. This segmentation is further refined by Radon transform. Details of seed selection through iterative voting and partitioning adjacent connected nuclei through radon transform are included below.

3.1 Seed Estimation with Iterative Voting

The basis for seeding (e.g., estimating centers of mass) is through geometric voting and perceptual grouping, and is implemented through the refinement of specifically tuned voting kernels [13]. In general, voting operates on the notion of continuity and proximity, which can occur at multiple scales, *e.g.,* points, lines, lines of symmetry, or generalized cylinders. The novelty of our approach is in defining a series of kernels that vote iteratively along the radial or tangential directions. Voting along the radial direction leads to localization of the center of mass, while voting along the tangential direction enforces continuity. At each iteration, the kernel orientation is refined until it converges to a single focal response. Voting kernels have a cone-shaped with an initial scale and spread (e.g., height and base) that is refined iteratively. These kernels are initially applied along the gradient direction, then at each consecutive iteration and at each grid location, orientation is aligned along the maximum local response. The method has excellent noise immunity, is tolerant to variations in target shape scale, and is applicable to a large class of application domains. Figure 2 shows a subset of voting kernels that vary in topography, scale, and orientation.

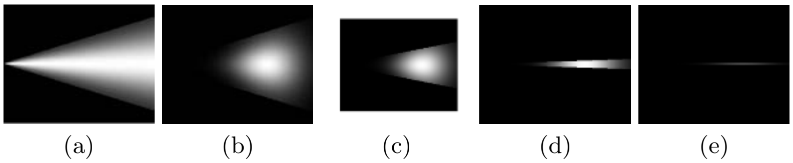


Fig. 2. Kernel topography: (a-e) Evolving kernel for the detection of radial symmetries (shown at a fixed orientation) has a trapezoidal active area with Gaussian distribution along both axes

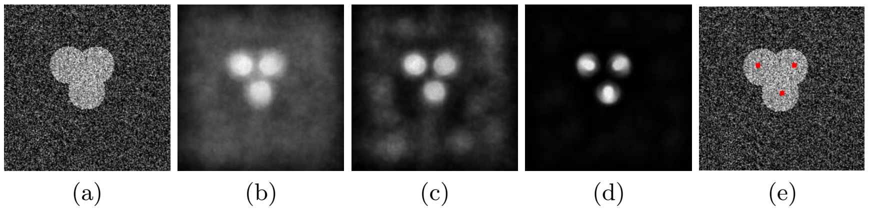


Fig. 3. Detection of radial symmetries for three overlapping blobs with with the signal-to-noise ratio of $6dB$: (a) original image; (b)-(d) the voting landscape at intermediate iterations; and (e) final localization following thresholding. The voting landscape is initially diffused in the background region, but it becomes more localized at the foreground in subsequent iterations.

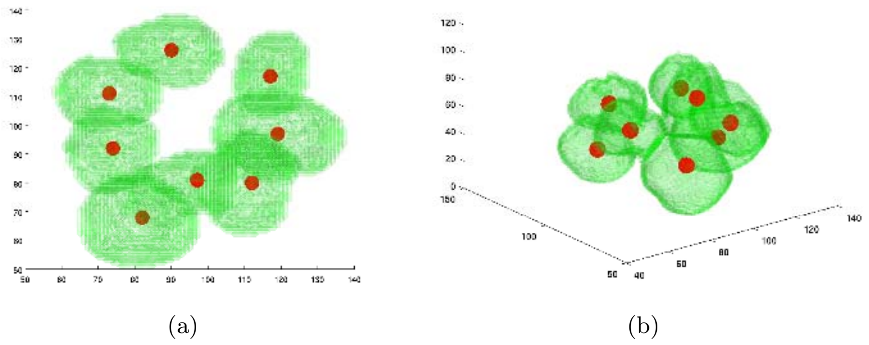
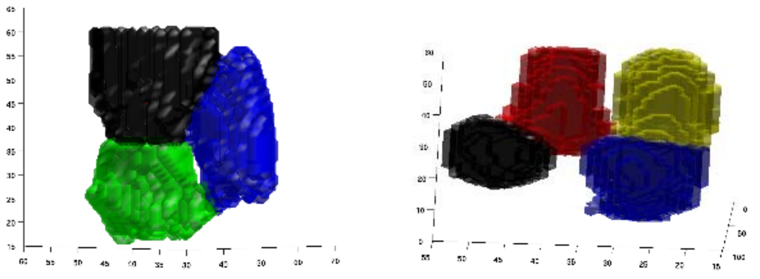


Fig. 4. Two views of voting results of a 3D clump of mammosphere: (a) top view; (b) side view

The iterative voting algorithm is presented for the 2D case in [9]. Our current implementation extends the 2D algorithm to volumetric data. An example of the application of radial kernels to overlapping 2D objects is shown in Figure 3 together with the intermediate results. The voting landscape corresponds to the spatial clustering that is initially diffuse and subsequently refined and focused into distinct islands. Figure 4 shows two views of the voting results for a 3D clump of mammosphere.

3.2 Partitioning of a Mammosphere from Seeded Nuclei

The process is initiated by a coarse segmentation of nuclei with a simplified 3D voronoi tessellation. Tessellation facilitates (1) identification of a local neighborhood where each nuclear region is contained within its own space, and (2) improved computational performance for each mammosphere prior to Radon transform. The first aspect has to do with constrained locality, which eliminates error and reduces ambiguities. Without tessellation, Radon transfer will fail because two neighboring nuclei may have a third nucleus that sits at the fold of the two touching nuclei. This condition is shown and visualized in Figure 5 from real data. The second point has to do with the fact that not all adjacent nuclei are connected and that there is a clear empty space between them. Under this condition, there is no need to refine the segmentation further.



(a) (b)

Fig. 5. A segmented example of the nuclear configuration where tessellation enforces locality: (a) the blue nuclei resides at the fold between the green and black nuclei and without an initial tessellation subsequent Radon transform refinement will fail; and (b) empty spaces between black and blue nuclei eliminates the need for Radon transform refinement

The details of Radon transform is as follows; however, for simplicity the 2D version is first described. The Radon transform represents an image as a collection of projections in a function domain $f(x, y)$ along various lines defined by the shortest distance ρ from the origin and the angle of inclination θ with the y axis:

$$R(\rho, \theta) = \int\int f(x, y)\delta(\rho - x\cos\theta - y\sin\theta)dxdy.$$

Properties of the Radon transform enable delineation of nearby touching objects. For example, two adjacent objects, represented by circles in Figure 6(a), and the corresponding Radon transform shown in Figure 6(b), has a local minimum at $\rho = 17$ and $\theta = 135°$. This local minimum corresponds to the integration over the line that separates the two objects with the smallest cross section.

Similarly, 3D Radon transform represents a 3D volume as a collection of projections in a function domain $f(x, y, z)$ along various planes defined by the shortest distance ρ from the origin, the angle of azimuth ϕ around the z axis and the angle of elevation θ around the y axis:

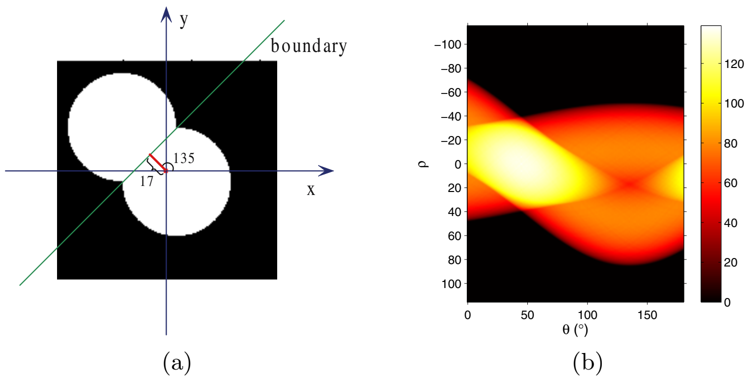


Fig. 6. An example of 2D object segmentation using the Radon transform: (a) synthetic object composed of two circles; and (b) corresponding Radon transform with local minimum at $\rho = 17$ and $\theta = 135°$

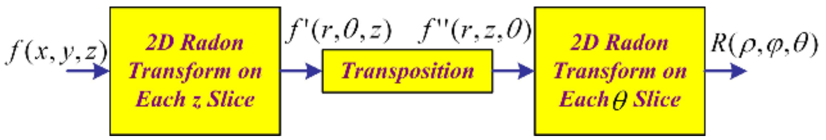

f(x, y, z)
2D Radon Transform on Each z Slice
f'(r, θ, z)
f''(r, z, θ)
Transposition
2D Radon Transform on Each θ Slice
R(ρ, φ, θ)

Fig. 7. The implementation of 3D Radon transform through 2D Radon transform

$$R(\rho, \phi, \theta) = \int\int\int f(x, y, z)\delta(\rho - x\cos\phi\cos\theta - y\sin\phi\cos\theta - z\sin\theta)dxdydz.$$

The Radon transform is a separable transform and its implementation is shown in Figure 7. A fast method for computing 3D radon transform via a direct Fourier method can be found in [3]. Given a local cube containing two nearby adjacent cells, each of which is bounded by a seed, the optimal plane separating these two cells should be located between the two seeds and have the smallest cross section. The local minimum in the 3D Radon transform corresponds to the integration over the optimal plane in the local cube.

4 Experimental Results

The proposed approach was implemented and applied to a set of samples that were imaged with deconvolution microscopy. The image resolution along the x and y directions is 0.15 *micron*, and the resolution along the z direction is 1.32 *micron*. Figure 8 shows several slices from a mammosphere, and the corresponding segmentation result is shown in Figure 9. Proposed method was implemented in C++ and Matlab with the average execution time of 2.5 minutes per volume data. The software was used to process 151 colonies, which has

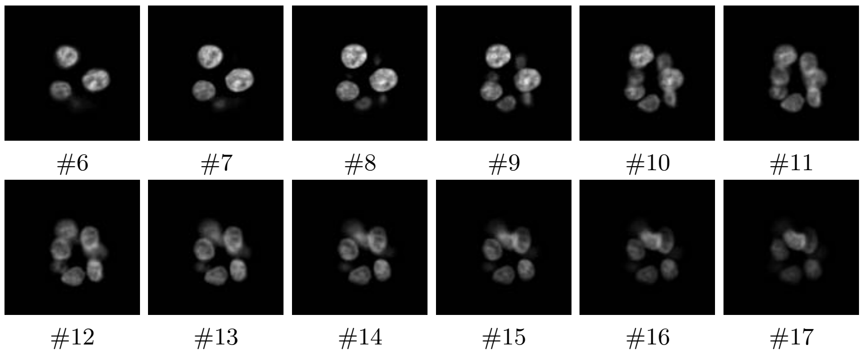


Fig. 8. Slices from a 3D cell colonv in the order of z direction

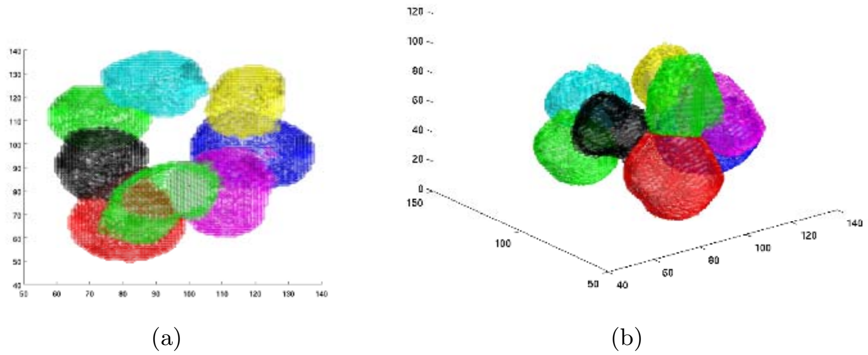


Fig. 9. Two views of final segmentation of a mammosphere for the stack shown in Figure 8: (a) top view; (b) side view

an average of 11 seeds per colony. 1771 seeds were estimated through iterative radial voting; however, 77 nuclei did not register any corresponding seeds, which indicates a detection error rate of 4%. This is presumably due to abnormal scale and shape of the nuclear volume with the following conditions:

- *Low contrast between overlapping nuclei:* Absence of gradient information between overlapping nuclei coupled with their accidental morphological properties provide ambiguous voting evidence that produces one fixed point instead of two.
- *Morphological abnormality:* Often a single nucleus has an abnormal elongated shape and radial voting merges multiple seed points into a single fixed point. This condition is highly correlated with previous case.
- *Incomplete information:* This is an imaging problem where imaging is incomplete and part of the nuclei is missing from the volumetric image, as shown in Figure 5.

– *Low sampling resolution in Z axis:* The current interpolation algorithm is linear for making an volumetric stack homogeneous in its X, Y, and Z dimension. Linear interpolation smooth the gradient in the Z direction and reduces contribution of the corresponding gradient information. An improved model will use some form of spline interpolation.

Finally, partitioning accuracy was compromised for 35 pair of overlapping nuclei from a total of 1850 pairs, which indicates an error rate of approximately 2%. These errors occur when the optimum planes for separating two nuclei is not the desired plane for partitioning two neighboring nuclei. The notion of desired planes has to do with those planes that bisect neighboring nuclei along points of maximum curvature. In this case, the error rate can be reduced through improved seed localization.

5 Conclusion and Future Work

This paper presented a series of geometric steps for segmentation of mammospheres from volumetric data. The first step localizes an approximation to center of mass for each nucleus and then partitioning clump of nuclei along minimal intersecting surfaces. Approximate solution to the center of mass is realized through iterative spatial voting, which is tolerant to variation in shape morphologies, perceptual surfaces, noise, and overlapping compartments. These centers of mass are then used to partition a clump of cells along minimal intersecting surfaces that are estimated by Radon transform. The technique has been tested on 151 colonies and their corresponding 3D volumes, and error rate is fully characterized. The system is being planned to be used for subsequent localization studies.

References

1. F. Attneave. Symmetry information and memory for patterns. *American Journal of Psychology*, 68:209–222, 1955.
2. D. Knowles, D. Sudar, C. Bator-Kelly, M. Bissell, and S. Lelievre. Automated local bright feature image analysis of nuclear protein distribution identifies changes in tissue phenotype. *Proceedings of National Academy of Science*, 103:4445–4450, 2006.
3. S. Lanzavecchia and P. Luigi Bellon. Fast computation of 3d radon transform via a direct fourier method. *Bioinformatics*, 12(2):212–216, 1998.
4. R. Malladi, J. Sethian, and B. Vemuri. Shape modeling with front propagation: a level set approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:158–175, 1995.
5. R. Murphy. Automated interpretation of subcellular locatoin patterns. In *IEEE Int. Symp. on Biomedical Imaging*, pages 53–56, April 2004.
6. C. Ortiz De Solorzano and et. al. Segmentation of nuclei and cells using membrane protein. *Journal of Microscopy*, 201:404–415, March 2001.

7. C. Ortiz De Solorzano, R. Garcia, A. Jones, D. Pinkel, W. Gray, D. Sudar, and S. Lockett. Segmentation of confocal microscope images of cell nuclei in thick tissue section. *Journal of Microscopy*, 193(3):193–212, 1999.
8. B. Parvin, Q. Yang, G. Fontenay, and M. Barcellos-Hoff. Biosig: An imaging bioinformatics system for phenotypic studies. *IEEE Transactions on Systems, Man and Cybernetics*, Part B, 33:814–824, 2003.
9. S. Raman, B. Parvin, C. Maxwell, and M. Barcellos-Hoff. Geometric approach segmentation and protein localization in cell cultured assays. In *Int. Symposium on Visual Computing*, pages 427–436, 2005.
10. A. Sarti, C. Ortiz De Solorzano, S. Lockett, and R. Malladi. A geometric model for 3-d confocal image analysis. *IEEE Transactions on Biomedical Engineering*, 47(12):1600–1610, December 2000.
11. V.M. Weaver, A.H. Fischer, O.W. Petersen, and M.J. Bissel. The importance of the microenvironment in breast cancer progression: recapitulation of mammary tumorigenesis using a unique human mammary epithelial cell model and a three-dimensional culture assay. *Biochemical Cell Biology*, 74(12):833–51, 1996.
12. Q. Yang and B. Parvin. Harmonic cut and regularized centroid transform for localization of subcelular structures. *IEEE Transaction on Biomedical Engineering*, 50(4):469–476, 2003.
13. Q. Yang and B. Parvin. Perceptual organization of radial symmetries. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 320–325, 2004.

Viewpoint Selection for Angiographic Volume

Ming-Yuen Chan, Huamin Qu, Yingcai Wu, and Hong Zhou

Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
{pazuchan, huamin, wuyc, zhouhong}@cse.ust.hk

Abstract. In this paper, we present a novel viewpoint selection framework for angiographic volume data. We propose several view descriptors based on typical concerns of clinicians for the view evaluation. Compared with conventional approaches, our method can deliver a more representative global optimal view by sampling at a much higher rate in the view space. Instead of performing analysis on sample views individually, we construct a solution space to estimate the quality of the views. Descriptor values are propagated to the solution space where an efficient searching process can be performed. The best viewpoint can be found by analyzing the accumulated descriptor values in the solution space based on different visualization goals.

1 Introduction

Direct volume rendering (DVR) is a powerful tool for angiographic volume as it can help clinicians analyze the anatomical structures present in the volume. Viewpoint selection is an important issue to the effectiveness of DVR. More 3D structures can be revealed at a proper viewpoint while severe occlusion and less informative results may be delivered at a bad viewpoint. It is time consuming for the viewers to find the optimal viewpoints by trial and error. Therefore, this paper addresses the problem by proposing a view-selection framework for angiographic volume based on some objective criteria.

As mentioned in different previous works [1] on viewpoint selection, it is subjective to judge the aesthetic value and appropriateness of a view. Different quantifications and measurements of visual information present in an image have been proposed. However, unlike typical view analysis, medical applications require a more objective and effective assessment on the accuracy and performance of the methodology. Therefore, we establish several objective criteria on the best view for angiograms, based on the common medical concerns of clinicians.

Even though a proper view evaluation can be derived from the medical considerations, optimal view searching is still a difficult problem. In order to find the best view, a considerable amount of computations are required to analyze different views in a huge search space. It is obvious that we cannot try out all possible views which are infinite in number. Assumptions are always exerted on the view selection (e.g., limited viewing angles) to restrict the range of the search space. Methods like canonical views [2] and normal clustering [3] have been suggested for aesthetic purposes, but are not always applicable to medical images.

Some researchers proposed to use different optimization methods to search for an optimal solution. The process is slow and may easily lead to local optimal solutions. Sampling of the view space and compromising with local optimal views are possible solutions. However, only a limited number of views are sampled for the sake of computation and the best view may still be missed.

In this paper, we design several view descriptors based on the typical concerns of clinicians to tackle the problems of visibility, self-occlusion and coverage of important structures in angiograms. After that, we propose a framework that projects those view descriptors onto a solution space where the viewpoint can be easily selected. The solution space can be constructed efficiently in one pass without evaluating the views individually.

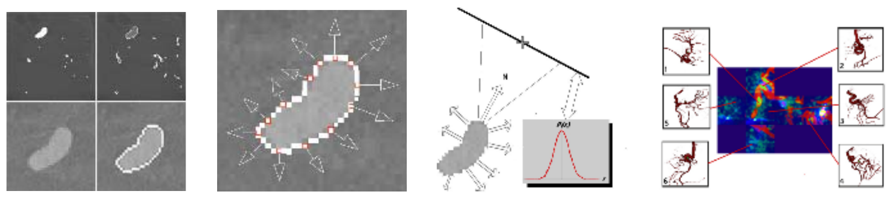
The rest of the paper is organized as follows. We introduce the previous work in Section 2 and describe the proposed framework for view selection in Section 3. The initialization process is covered in Section 4. The details about view descriptors are explained in Section 5. The projection process is explained in Section 6. The solution space and view selection are explained in Section 7. Experimental results and conclusion are covered in Section 8 and 9.

2 Previous Work

Viewpoint selection is a challenging problem which has been studied extensively in various fields, like computer graphics [4] and computer vision [5]. A proper viewpoint can bring a meaningful and informative view to viewers. Several methods have been introduced to evaluate the quality of views using different metrics and a recent survey can be found in [1]. Methodologies like surface area entropy [6], curvature entropy [7], entropy of semantic parts have been studied. Vázquez et al. [5] proposed a view selection method based on view entropy and Sbert et al. [8] suggested the use of Kullback-Leibler distance as a view quality measure. Lee et al. [9] also proposed a view selection method based on view saliency. Most of the works focus on mesh-based data and the evaluation is based on the geometries present in the data. However, some measures, like visibility, are not clearly defined in volume data where geometry is usually not present. Semi-transparent structures present in the image may be considered as meaningful characteristic features in volume rendering. Previous works on view selection for volume data are scant. Recently, Bordoloi and Shen [10] suggested the use of a voxel-based entropy function as a goodness measure of viewpoints. Takahashi et al. [11] proposed to decompose a volume into feature components and a best view is computed by compromising between different local optimal viewpoints. Some related approaches can also be found in [12].

3 Overview

We propose a framework for view selection by propagation of view descriptors on a solution space and the pipeline is shown in Figure 1. During the initialization process, *regions of interest* (ROI) and the corresponding boundaries are defined by users. Values are assigned to the view descriptors of each voxels. The propagation of the view descriptors is performed by projecting the values along different directions onto the boundaries of the volume, which constitute the final solution space. Optimal views can be found by searching in the solution space which is represented as a color-encoded projection map. To deliver proper results that conform with clinical requirements, we proposed several view descriptors for angiographic visualization, namely *visibility*, *coverage* and *self-occlusion*.



(a) (b) (c) (d)

Fig. 1. Viewpoint selection pipeline: (a) selecting ROIs and boundaries; (b) initializing the view descriptors; (c) propagating view descriptors to the boundaries; (d) selecting viewpoints from the projection maps

4 Initialization

In the conventional approaches, the objects of interest are always assumed to be located at the center. Instead, we allow viewers to define the region of interest in the volume such that a good view can be found accordingly. The viewpoint is placed on a view sphere (i.e., a fixed distance away from the ROI) while the exact position depends on the view evaluation result.

To ease the difficulty in the user selection process in angiograms, a MIP-guided selection method [13] can be used to define regions of interest (i.e., voxels to be visible in the final view) by selecting the projected voxels in a MIP. Besides, the user can label those voxels in the slices of the volume. For simplicity, the voxels of interest can be specified using the histogram. After selecting the voxels of interest, a bounding region is generated automatically. The region encapsulates all the selected voxels and the voxels on the boundary of the region will become the starting points of the projection process. The simplest way is to construct a rectangular bounding box.

However, to ensure a proper and efficient projection, the region should be a tight minimum bound. This minimizes the initialization cost and increases the significance of the projected result. A bounding region may be too large if the selected points are sparse (e.g., irregular structures and branches of vessels) in the volume. To solve this problem, clustering is performed on the selected points and smaller regions are formed. A large vessel tree may be clustered into smaller branches. The projection process will then be conducted on each cluster individually. As the propagation is partially driven by the shape of the objects of interest for determining the normal direction for projection, it is important to select appropriate points on the boundary of the objects to preserve the shape.

The voxels on the bounding region are the starting points of the propagation process. Each of them is assigned a set of initial values of view descriptors (see Section 5). In our experiment, we initialize values for visibility by performing ray-casting from each boundary voxel to all the selected points. The initial value is given by the average of the accumulated opacity of the rays. It estimates the obstruction caused by the voxel and indicates the visibility within the bounding region at the voxel. Besides, a value of coverage is assigned to each initial boundary voxel according to the importance given by the user. If no importance values are given, every voxel is assigned a constant value.

5 View Descriptors

View descriptor is a term used to describe the criterion employed in the view evaluation process. The choice of the descriptors depends on the visualization goal and also the effectiveness of the measures. In our proposed framework, we adopt the visibility, coverage and self-occlusion as they are the practical criteria for a clinical view on angiograms. They can be applied on volumetric data and are tailored for angiographic purposes.

The visibility descriptor, similar to the visibility ratio, represents the effect of one object in space blocking another object from view. However, unlike the surface-based model, the value indicates the level of obstruction of voxels in the volume data. We assign an occlusion value to each voxel according to the transfer function. A more opaque voxel has a higher occlusion value. In the propagation process, the accumulated occlusion values projected onto the solution space indicate the visibility of the views. The coverage descriptor, on the other hand, indicates the exposure of the object to the view. In a surface-based model, it is easily found by considering the area of surfaces visible to the view. In the volume data, we try to maximize the number of boundary voxels visible to the view. Besides, we propose another descriptor to indicate the degree of self-occlusion. In a volume data with complicated anatomies, the structures (e.g., vessels) may obscure each other in the view. The value of the self-occlusion descriptor tries to indicate the possible occlusion existing in the view.

Actually, our framework does not exert any restriction on the use of view descriptors. More complicated and high-level semantic descriptors can be used to suit different applications. However, the above descriptors can satisfy the typical requirements of clinicians for angiographic purposes.

To evaluate the quality of a view, we establish a view measure which consists of a composite value of the view descriptors. Instead of sampling the view space and evaluating the views one by one, we reverse the process and project the values to the solution space. An estimated global optimal view can be found efficiently by searching in the solution space. Different from the typical approaches which make assumptions on the orientation of the structures in the volume and only select views in a certain range of directions, our method allows the search space to span all possible directions. It is more reasonable for medical images in which hidden pathologies may exist at different positions and orientations.

5.1 Visibility

The optimal view should attain the highest visibility and the region of interest should not be occluded by irrelevant structures. Therefore, we design a view descriptor to indicate the degree of occlusion caused by the context. Typically, the visibility problem can be solved easily for surface-based models. However, such surfaces are not defined in volume data and the occlusion is due to the opaque voxels in the context. Therefore, each opaque voxel is assigned an occlusion value and is propagated to the solution space (Figure 2(b)). As the occlusion effect is more severe if the voxels locate in between the viewpoint and the ROI in the normal direction, the projection direction always coincides with the normal of the ROI. The viewpoints at the normal receive the highest projected value while those views farther way from it receive a lower value. The projection is performed on all the contextual voxels and the values accumulate at the boundary planes (solution space). The higher the accumulated value, the higher the occlusion level. Therefore, a good view should keep this value at a minimum or zero.

5.2 Coverage

The quality of the view may not be good even if it attains a high degree of visibility. For instance, an unoccluded view may only reveal a small portion of the ROI. Besides, there may be many candidate views which have high visibility, therefore, we should have an additional criterion to evaluate the goodness of a view. Following the basic idea of surface area entropy [6] in a polygonal scene, we propose a descriptor for coverage of ROI in volume data which can be then propagated from the boundary of the ROI (Figure 2(c)). Users may specify certain regions that they are interested in and assign different coverage values according to the importance. After that, the value will be propagated, in a similar manner, along the nomral direction of the ROI.

5.3 Self-occlusion

Angiograms are usually cluttered and fuzzy in nature. Self-occlusion between the vessels within the ROI usually exists and should be minimized. However, it is infeasible to perform a surface-based visibility test on such a complicated scene. Therefore, we assign occlusion values to different parts of the object according to their opacity and then compute the mutual occlusion effect among them.

First, we build an octree of the angiographic volume. The voxels of interest are labeled and structures are encoded into the nodes of the octree. Then, we perform clustering based on the similarity of the neighboring voxels (e.g., opacity or other high-level feature measures). Smaller nodes are clustered into larger nodes for the sake of efficient computation. A vessel tree structure is represented by a series of nodes of different sizes (Figure 2(a)). To analyze the possible self-occlusion, we project each node to the projection plane in all directions from the node to other nodes (Figure 2(d)). Similar to the previous case, the projection value and the kernel depend on the size of the node, opacity, distance, etc. The running time of the process is $O(n^2)$, where n is the number of nodes. For the sake of performance, we limit the number of nodes by pruning the tree and ignoring the nodes of smaller size. This does not affect the final result much as the occlusion effect of an isolated voxel (e.g., noise) is insignificant.

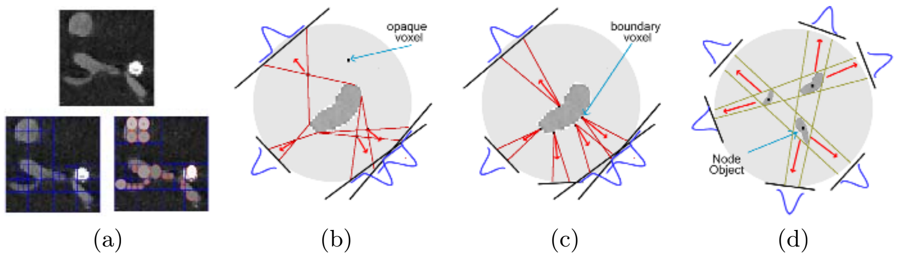


Fig. 2. View descriptor projection: (a) octree representation; (b) visibility projection; (c) coverage projection; (d) self-occlusion projection

6 Projection

After defining the region of interest and initializing the corresponding descriptor values, a propagation process is performed by projecting the values onto the view solution space. One major assumption of the projection process is that the best views are always located at the front of the object (i.e., normal direction). This results in a high projected value for those views. The views with larger incident angle (i.e., deviated from the normal direction to the object) should receive a lower value. The projection depends on the orientation and distance of each voxel from the viewpoint. During the projection, we try to propagate the effect of descriptors exerted by the voxels to a solution space and the projected values on the solution space should be an accumulated effect from different voxels. This method is different from the conventional approaches which try to compute the accumulated effects on each view individually. The values are projected onto the boundaries of the volume. The six boundary planes, in the case of a cubic volume data, become the solution space for view analysis. Each pixel on the plane actually represents a viewpoint at a certain viewing direction from a fixed distance to the center of the region of interest.

Instead of projecting values at all directions, we only project at the normal directions of the ROIs. This is based on the previous assumption on the best view. The opposite direction to the normal represents a back face to the view. To avoid unnecessary projections and increase the efficiency of the process, only the normal direction should be considered for each projection. Actually, the projection is similar to splatting and the projected region depends on the position of the source, the projection plane and also the parameters used. Unlike unidirectional splatting, our projection direction also depends on the shape of the object. Although the projection kernel should take the shape of the object into account for the highest accuracy, a uniform gaussian kernel is used as it can relieve the effort for the shape computation which is view dependent. This gives satisfactory results in our experiments as all the voxels contribute to the final result, and the shape can be preserved in a large extent during the propagation.

7 Solution Space and Viewpoint Selection

The propagation process terminates at the boundary of the volume. Each pixel in the boundary planes receives values of descriptors from different projection rays passing through it. The boundary voxels with the projected values constitute the solution space for view selection. The feature vector (view measure) of a boundary voxel can be represented as

$$\boldsymbol{v}(x) = \{v_1, v_2, \ldots, v_k\} \qquad where \;\; v_i = \frac{1}{n}\sum_{j=0}^{n} P_i(j, x) \tag{1}$$

v_i represents the averaged projected value of descriptor i at position x. The feature values accumulate the contributions of different voxels and each voxel propagates view effects to different views in the solution space. By analyzing the values in the solution space, good views can be selected according to different visualization goals.

To allow users to select a proper view according to their preferences, we display the feature values in the solution space by unfolding it into a plane, which we refer to as projection map, to show the results of the propagation. To visualize the feature vector of the map elements, the values of the view descriptors are encoded into different color channels (e.g., RGB), as shown in Figure 3. The vector values can also be integrated into a composite value as

$$I = \sum_{i=0}^{k} \alpha_i v_i \tag{2}$$

where α_i is the weight given to the descriptor v_i (it may be negative for some descriptors like visibility and occlusion to penalize such effects). The weights can be learnt using machine learning approaches or determined by users according to their visualization concerns. The composite value can indicate the quality of the view at a specific viewpoint given by the corresponding boundary voxel location. To increase the stability of the view, we apply a gaussian filter to the map.

For a volumetric dataset of size $256 \times 256 \times 256$, the map size is 393216 and each element of the map represents a candidate view. After the propagation process, each element should have a vector of accumulated view descriptor values. The view space can be interpreted as an energy function of view descriptors

$$f(x) = \alpha_1 v_{vis}(x) + \alpha_2 v_{cov}(x) + \alpha_3 v_{self}(x) \tag{3}$$

The optimal views can be found easily using the gradient decent-based algorithm or manual selection. As the propagation is completed in one pass, the solution space can be precomputed and all the local or even global optimals can be obtained efficiently. Due to the complex vascular structures in the images, the view selection process can easily lead to a local optimal solution. Our method avoids this problem by sampling at a much higher rate and thus a global optimal solution is more likely to be found.

8 Experimental Results

We test our method with several angiographic datasets ($256 \times 256 \times 256$) with vascular tumors. Our method is used to select proper viewpoints for diagnosis. The optimal view should have: 1) high visibility such that the tumor can be seen clearly without being occluded by the vessels; 2) high coverage such that more vessels and contextual information can be found; 3) minimum self-occlusion such that the vessels are not obscuring each other.

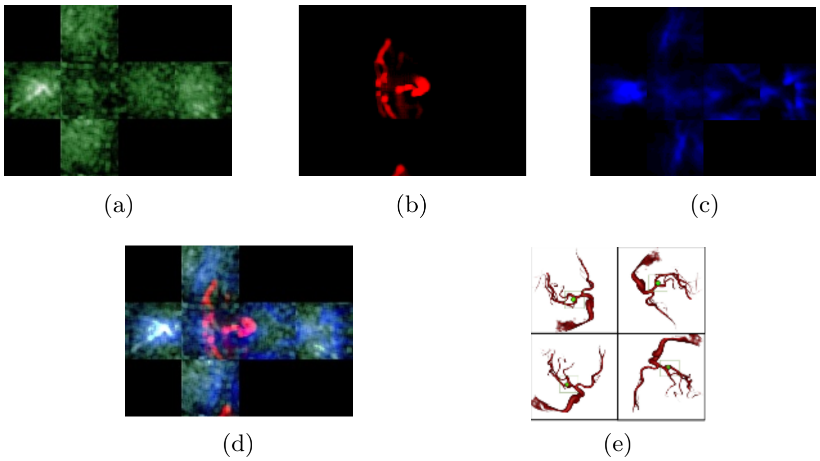


Fig. 3. Experiment on angiographic data ($256 \times 256 \times 256$): (a) coverage map; (b) visibility map; (c) self-occlusion map; (d) combined projection map; (e) selected views

Figure 4 shows the result of our experiment. To demonstrate the effectiveness of our method, several views are selected from the regions with high and low values in the maps. We can see that the tumor (green in color) is always occluded by other vascular structures in the views with high visibility value. However, among those unoccluded views, certain views may reveal more information about the contextual structures. Therefore, we use coverage as a criterion to select the views which can reveal more vessels. Figure 4(d) shows that more vessels are revealed for a higher coverage value. This can be reflected in the higher

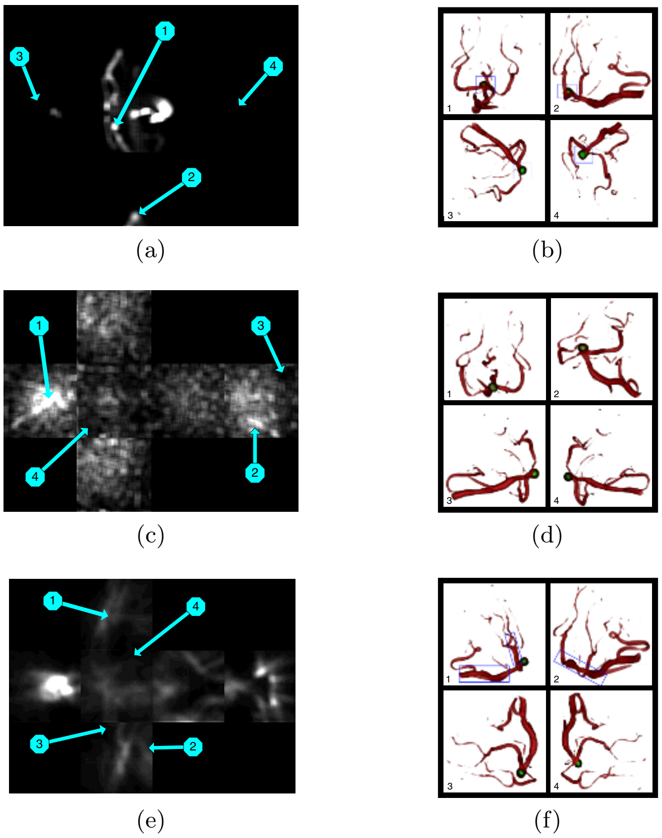


Fig. 4. Projection maps of visibility (a), coverage (c), self-occlusion (e) and some selected views from the maps (b)(d)(f)

occupancy ratio of vessels in the images. However, the large vessels near the tumor may not be seen clearly at certain angles due to the self-blockage. To minimize the occlusion effect among the vessels, the self-occlusion value is also considered, as shown in Figure 4(e). This shows that some viewpoints with high coverage may have a severe occlusion effect. A better view can be achieved by selecting a viewpoint with a smaller self-occlusion effect. Figure 3 shows the projection maps of visibility, coverage and self-occlusion. Some views are selected by considering the above criteria.

9 Conclusion

In this paper, we presented a framework for viewpoint selection for angiographic volume. View descriptors for visibility, coverage and self-occlusion of important structures in the data are established to address the typical concerns of clinicians. By projecting the view descriptor values onto the projection maps, optimal viewpoints can be efficiently found by searching in the solution space. The effectiveness of our method is demonstrated in the experiments on different angiograms. Compared with conventional methods, our method can deliver a more accurate solution by sampling at a much higher rate with less computation.

Acknowledgements

This work has been supported by HK RGC grant CERG 618705. The dataset is courtesy of Viatronix Inc., USA. We would like to thank the anonymous reviewers for their valuable comments.

References

1. Polonsky, O., Patanè, G., Biasotti, S., Gotsman, C., Spagnuolo, M.: What's in an image? The Visual Computer **21** (2005) 840–847
2. Blanz, V., Tarr, M., Bulthoff, H.: What object attributes determine canonical views? Perception (1999) 579–599
3. Kumar, S., Manocha, D., Garrett, B., Lin, M.: Hierarchical back-face culling. In: 7th Eurographics Workshop on Rendering. (1996) 231–240
4. Kamada, T., Kawai, S.: A simple method for computing general position in displaying three-dimensional objects. Comp. Vis. Graph. Image Proc. **41** (1988) 43–56
5. Vázquez, P., Feixas, M., Sbert, M., Heidrich, W.: Automatic view selection using viewpoint entropy and its application to image-based modelling. Computer Graphics Forum **22** (2004) 689–700
6. Vázquez, P., Feixas, M., Sbert, M., Heidrich, W.: Viewpoint selection using viewpoint entropy. In: Vision Modeling and Visualization Conference. (2001) 273–280
7. Page, D.L., Koschan, A., Sukumar, S.R., Roui-Abidi, B., Abidi, M.A.: Shape analysis algorithm based on information theory. In: ICIP (1). (2003) 229–232
8. Sbert, M., Plemenos, D., Feixas, M., Gonzalez, F.: Viewpoint quality: Measures and applications. In: Computational Aesthetics in Graphics, Visualization and Imaging. (2005) 185–192
9. Lee, C.H., Varshney, A., Jacobs, D.: Mesh saliency. ACM Transactions on Graphics **24** (2005) 659–666
10. Bordoloi, U., Shen, H.W.: View selection for volume rendering. In: IEEE Visualization. (2005) 487–494
11. Takahashi, S., Fujishiro, I., Takeshima, Y., Nishita, T.: A feature-driven approach to locating optimal viewpoints for volume visualization. In: IEEE Visualization. (2005) 495–502
12. Pommert, A., Bomans, M., Hohne, K.H.: Volume visualization in magnetic resonance angiography. IEEE Computer Graphics and Applications **12** (1992) 12–13
13. Chan, M.Y., Wu, Y., Qu, H., Chung, A.C.S., Wong, W.C.K.: Mip-guided vascular image visualization with multi-dimensional transfer function. In: Computer Graphics International. (2006) 372–384

Recognizing Action Primitives in Complex Actions Using Hidden Markov Models

V. Krüger

Aalborg Media Lab
Aalborg University, Copenhagen
Lautrupvang 15
2750 Ballerup
Denmark
vok@media.aau.dk

Abstract. There is biological evidence that human actions are composed out of action primitives, similarly to words and sentences being composed out of phonemes. Given a set of action primitives and an action composed out of these primitives we present a Hidden Markov Model-based approach that allows to recover the action primitives in that action. In our approach, the primitives may have different lengths, no clear "divider" between the primitives is necessary. The primitive detection is done online, no storing of past data is necessary. We verify our approach on a large database. Recognition rates are slightly smaller than the rate when recognizing the singular action primitives.

1 Introduction

There is biological evidence that actions and activities are composed out of action primitives similarly to phonemes being concatenated into words [23, 8, 22].

In this sense, one can define a hierarchy of *action primitives* at the coarsest level, and then *actions* and *activities* as the higher abstract levels where actions are composed out of the action primitives while activities are, in turn, a composition of the set of actions [2, 17][1]. It is an open problem how to define and detect these action primitives. It might probably be reasonable to assume that a particular set of action primitives can only be defined in context of the specific application at hand.

In this paper we will deal with the recovery of the sequence of the action primitives out of an action, when a set (or alphabet) of action primitives is given.

In other words, if we have given an alphabet of action primitives P and if we define a particular *action* to be a sequence $S = a_1a_2a_3 \ldots a_T$ of these action primitives, then it is our interest to recover these primitives and their precise order. This problem is closely related to speech recognition where the goal is

[1] In the following, we define the term *action* as a sequence of action primitive of arbitrary length.

to find the right sequences of phonemes (see Sec. 2). Once we have parsed and detected the sequence of action primitives in the observed sequence, this sequence of action primitives could identify the action. (In speech recognition, the sequence of detected phonemes is used to identity the corresponding word.)

One possibility to recognize the action is to define an *action-grammar* for each action, based on the action primitives as the alphabet and to use a parsing approach for recognition, as suggested in [25, 13].

In order to take into account possible noise and imperfect data, we base our approach on Hidden Markov Models (HMMs) [10, 18] and represent our action primitives with HMMs.

Thus, given a set of action primitives where each action primitive is represented by an HMM and given an observed sequence S of these action primitives where

1. the order of the action primitives and
2. the duration of each single action primitive and the position of their boundaries

are unknown, we would like to identify the most likely sequence of action primitives in the observation sequence S for subsequent parsing.

According to the biological findings, the representation for action recognition is closely related to the representation for action synthesis (i.e. the motor representation of the action) [23, 8, 22]. This motivates us to focus our considerations in this paper to actions represented in joint space. Thus, our actions are given as sequences of joint settings. A further justification for this approach is that this action representation can then be used, in future work, to bias 3D body trackers as it operates directly on the 3D parameters that are to be estimated by the 3D tracker. The focus of this paper on joint data is without limiting generality. In our on-going research we have applied the techniques of this paper also action recognition based on silhouettes.

This paper is structured as follows: In Sec. 2 will will give an overview of related work. In Sec. 3 we will discuss our approach for the HMM-based recognition of the action primitives. In Sec. 4 we present our extensive experimental results. The paper is concluded then in Sec. 5 with final comments.

2 Related Work

The recovery of phonemes in speech recognition is a closely related to our problem. In speech recognition, acoustic data gets samples and quantized, followed by using the LPC (Linear Predictive Coding) to compute a *cepstral* feature set, or by a PLP (Perceptual Linear Predictive) analysis [9]. In a later step, time slices are analyzed. Gaussians are often used to compute likelihoods of the observations of being a phoneme [11]. An alternative way is to analyze time slices with an Artificial Neural Network [3]. Timeslices seem to work well on phonemes that have a very short duration. In our case, however, the action primitives have usually a much longer duration and one would have a combinatorial problem when considering time slices.

When viewing other agents performing an action, the human visual system seems to relate the visual input to a sequence of motor primitives. The neurobiological representation for visually perceived, learned and recognized actions appears to be the same as the one used to drive the motor control of the body [23, 8, 22]. These findings have gained considerable attention from the robotics community [24, 7]. In *imitation learning* the goal is to develop a robot system that is able to relate perceived actions to its own motor control in order to learn and to later recognize and perform the demonstrated actions.

In [15,14], Jenkins *et al.* suggest applying a spatio-temporal non-linear dimension reduction technique on manually segmented human motion capture data. Similar segments are clustered into primitive units which are generalized into parameterized primitives by interpolating between them. In the same manner, they define action units ("behavior units") which can be generalized into actions. In [12] the problem of defining motor primitives is approached from the motor side. They define a set of nonlinear differential equations that form a control policy (CP) and quantify how well different trajectories can be fitted with these CPs. The parameters of a CP for a primitive movement are learned in a training phase. These parameters are also used to compute similarities between movements. In [5, 1, 4] a HMM based approach is used to learn characteristic features of repetitively demonstrated movements. They suggest to use the HMM to synthesize joint trajectories of a robot. For each joint, one HMM is used. In [5] an additional HMM is used to model end-effector movement. In these approaches, the HMM structure is heavily constrained to assure convergence to a model that can be used for synthesizing joint trajectories.

Generally, there is a very large body of literature on action recognition. However, only a small subset is concerned with action primitives and their detection and recognition. In [26], Vecchio and Perona employ techniques from the dynamical systems framework to approach segmentation and classification. System identification techniques are used to derive analytical error analysis and performance estimates. Once, the primitives are detected an iterative approach is used to find the sequence of primitives for a novel action. In [16], Lu *et al.* also approach the problem from a system theoretic point of view. Their goal is to segment and represent repetitive movements. For this, they model the joint data over time with a second order auto-regressive (AR) model and the segmentation problem is approached by detection significant changes of the dynamical parameters. Then, for each motion segment and for each joint, they model the motion with a damped harmonic model. In order to compare actions, a metric based on the dynamic model parameters is defined. In [15, 14], Jenkins *et al.* suggest applying a spatio-temporal non-linear dimension reduction technique on manually segmented human motion capture data. Similar segments are clustered into primitive units which are generalized into parameterized primitives by interpolating between them. In the same manner, they define action units ("behavior units") which can be generalized into actions. While most scientists concentrate on the action representation by circumventing the vision problem, [19] takes a vision-based approach. They propose a view-invariant representation of action based on *dynamic instants* and *intervals*. Dynamic instants are used as primitives of actions which are computed from discontinuities of 2D hand trajectories. An interval represents the time period between two dynamic instants (key poses). A similar approach of using meaningful instants in time is proposed by Reng *et al.* [21] where key poses are found based on the curvature and covariance of the normalized trajectories. In [6] key poses are found through evaluation of anti-eigenvalues.

3 Representing and Recognizing Action Primitives Using HMMs

In order to approach the action recognition problem, we model each of the action primitives $P = \{a^1, a^2, \dots, a^N\}$ with a mixture-HMM where each observation function is a continuous Gaussian mixture with $M \geq 1$ mixtures. The mixture HMMs are trained based on demonstrations of a number of individuals. The Gaussian mixture are able to represent the variability across individuals to allow some degree of invariance across different individuals. The training results into a set of HMMs $\{\lambda_i | i = 1 \dots N\}$, one for each action primitive.

Once each action primitive is represented with an HMM, the primitives can generally simply be recognized with the classical recognition technique for HMMs by employing a maximum likelihood or a maximum a-posteriori classifier: Given an observation sequence O_t of an action primitive, and a set of HMMs λ_i, the maximum likelihood (ML)

$$\max_i P(O_t|\lambda_i) \tag{1}$$

identifies the most likely primitive. An alternative to the ML technique is the maximum a-posteriori (MAP) estimate that allows to take in to account the likelihood of observing each action primitive:

$$\max_i P(\lambda_i|O_t) = \max_i P(O_t|\lambda_i)P(\lambda_i) \ , \tag{2}$$

where $P(\lambda_i)$ is the likelihood that the action, represented by the HMM λ_i appears.

3.1 Recognition with HMMs

In general, the likelihood of an observation for some HMM λ_i can be computed as

$$\begin{aligned} P(O|\lambda_i) &= \sum_S P(O, S|\lambda_i) &(3)\\ &= \sum_S P(O|S, \lambda_i)P(S|\lambda_i) &(4)\\ &= \sum_S \prod_{t=0}^{T} P(O_t|S_t, \lambda_i) \prod_{t=0}^{T} P(S_t|S_{t-1}, \lambda_i) \ . &(5)\end{aligned}$$

Here, one marginalizes over all possible state sequences $S = \{S_0, \dots, S_T\}$ the HMM λ_a can pass through.

To apply this technique to our problem directly is difficult as we would need to know *when* to evaluate, i.e. at what time steps t we should stop and do the maximum-likelihood estimation to find the most likely action primitive that is just now being observed.

Instead of keeping the HMMs distinct, our suggestion is to insert the "action" a of the HMM λ_a as a random variable into Eq. (5) and to rewrite it as

$$P(O|a) = \sum_S \prod_{t=0}^{T} P(O_t|S_t, a_t) P(S_t, a_t|S_{t-1}, a_{t-1}) \ . \tag{6}$$

In other words, we would like to estimate at each time step the action a and the state S from the previously seen observations, or, respectively, the probability of λ_a being a model of the observed action:

$$P(S_T, a_T|O_{0:T}) = \prod_{t=0}^{T} P(O_t|S_t, a_t) P(S_t, a_t|S_{t-1}, a_{t-1}) \tag{7}$$

The difference in the interpretation becomes more clear when we write Eq. (7) in a recursive fashion:

$$P(S_{t+1}, a_{t+1}|O_{0:t+1}) = P(O_{t+1}|S_{t+1}, a_{t+1}) P(S_{t+1}, a_{t+1}|S_t, a_t) P(S_t, a_t|O_{0:t}) \tag{8}$$

This is the classical Bayesian propagation over time. It computes at each time step t the likelihood of observing the action a_t while having observed $O_{0:t}$. If we ignore the action variable a_t, then Eq. (8) explains the usual efficient implementation of the forward algorithm [10]. Using the random variable a_t, Eq. (8) defines a pdf across the set of states (where the state vector S_t is the concatenation of state vectors of each individual HMM) and the set of possible actions. The effect of introducing the action a might not be obvious: using the action a, we do not any more estimate the likelihood of an observation, given a HMM λ_a. Instead, we compute *at each time step* the probability mass function (pmf) $P(S_t, a_t|O_{0:t})$ of each state and each identity, given the observations. By marginalizing over the states, we can compute the pmf $P(a_t|O_{0:t})$ for the action at each time step. The likelihood $P(a_t|O_{0:t})$ converges to the most likely action primitive as time progresses and more data becomes available (see Fig. 1). From Fig. 1 it is apparent that the pmf $P(a_t|O_{0:t})$ will remain constant after convergence as one action primitive will have the likelihood 1 and all other primitive likelihoods have vanished. To properly evaluate the entire observation sequence, we apply a voting scheme that counts the votes after each convergence and then restarts the HMMs. The states are initialized with the present observation likelihoods and then propagated with the transition matrix as usual. Fig. 2 shows the repeated convergence and the restarting of the HMMs. In the example shown in Fig. 2 we have used two concatenated action primitives, denoted by the green curve with the "+" and by the blue curve with the "o", respectively. The first action

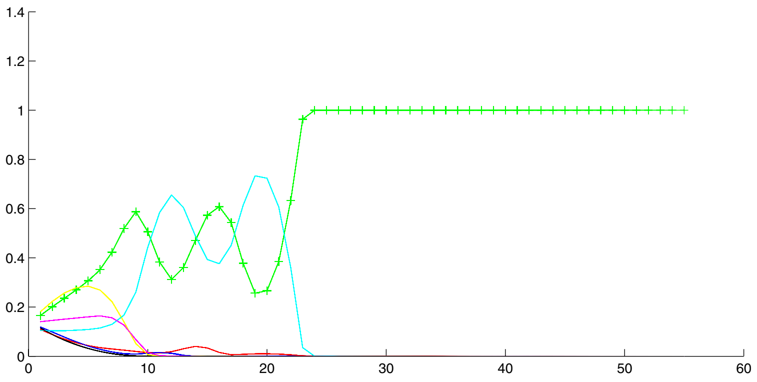


Fig. 1. Shows an example for a typical behavior of the pmf $P(a_t|O_{0:t})$ for each of the actions a as time t progresses. One can see that the likelihood for one particular action (the correct one in this example, marked with "+") converges to 1 while the likelihoods for the others vanish.

primitive was in the interval between 0 and 51, while the second action primitive was from sample 52 to the end. One can see that the precise time step when primitive 1 ended and when primitive 2 started cannot be identified. But this does not pose a problem for our recovery of the primitives as for us the order matters but not their precise duration. In Fig. 1 a typical situation can be seen where the observed data did not give enough evidence for a fast recognition of the true action.

4 Experiments

For our experiments, we have used our MoPrim [20] database of human one-arm movements. The data was captured using a **FastTrack** Motion capture device with 4 electromagnetic sensors. The sensors are attached to the torso, shoulder, elbow and hand (see Fig. 3). Each sensor delivers a $6D$ vector, containing $3D$ position and $3D$ orientation thus giving a $24D$ sample vector at each time-step (4 sensors with each $6D$). The MoPrim database consists of 6 individuals, showing 9 different actions, with 20 repetitions for each. The actions in the database are simple actions such as *point forward*, *point up*, *"come here"*, *"stop!"*. Each sequence consists of $\approx$ 60-70 samples and each one starts with 5 samples of the arm in a resting position where it is simply hanging down.

Instead of using the sensor positions directly, we transform the raw $24D$ sensor data into joint angles: one elbow angle, one shoulder angle between elbow, shoulder and torso and a 3D orientation of the normal of this shoulder-elbow-torso-triangle. The orientation of the normal is given with respect to the normal of this triangle when the arm is in resting position. All angles are given in radians. No further processing of the MoPrim data was done.

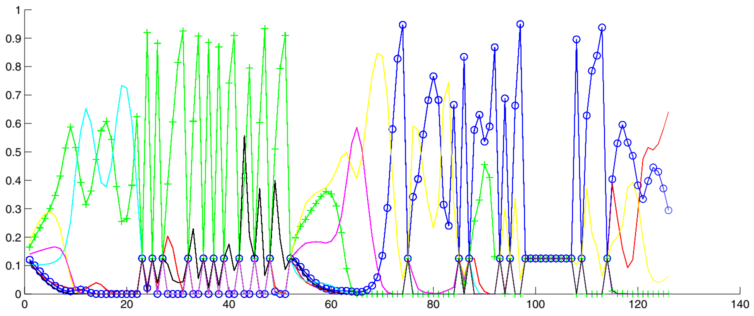


Fig. 2. Shows an example for a typical behavior of the pmf $P(a_t|O_{0:t})$ as time t progresses. The input data consisted of two action primitives: first, action primitive "2", marked with "+", then, action primitive "3", marked with "o". One can see that until $\approx$ sample 52 the system converges to action "2", after sample 70, the system converges to primitive 3. The length of the first sequence is 51 samples, the length of sequence 2 is 71 samples.

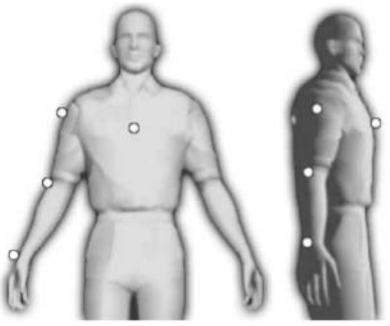


Fig. 3. Marks the positions of the magnetic sensor on the human body

We have carried out several different experiments:

1. In the first test, we tested for invariance with respect to the performing human. We have trained nine HMM for nine action. Each of the HMMs was trained on 6 individuals and all the 20 repetitions of the actions. The recognition testing was then carried out on the remaining individual (leave-one-out-strategy). The HMMs we use were mixture HMMs with 10 states and 5 mixtures per state.
2. In this test, we tested for invariance with respect to the variations within the repetitions. We have trained nine HMMs for nine actions. Each HMM was trained on all individuals but only on 19 repetitions. The test set consisted of the 20th repetition of the actions.
3. As a base line reference, we have tested how good the HMMs are able to recognize the actions primitives by testing action primitive sequences of length 1. Here, the HMMs were trained as explained under 2 above. This test reflects the recognition performance of the classical maximum-likelihood approach.

4. We have repeated the above three experiments after having added Gaussian noise with zero mean and a standard deviation of $\sigma = 0$, $\sigma = 0.3$ and $\sigma = 1$ to the training and testing data. As all angles are given in radians, thus, this noise is considerable.

To achieve a good statistic we have for each test generated 10.000 test actions of random length ≤ 100. Also, we have systematically left out each individual (action) once and trained on the remaining ones. The results below are averaged across all leave-one-out tests. In each test action, the action primitives were chosen randomly, identically and independently. Clearly, in reality there is a strong statistical dependency between action primitives so that our recognition results can be seen as a lower bound and results are likely to increase considerably when exploiting the temporal correlation by using an action grammar (e.g. another HMM).

The results are summarized in Table 1. One can see that the recognition rates of the individual action primitives is close to the general base-line of the HMMs. The recognition rates degrade with increasing noise which was to be expected, however, the degredation effect is the same for all three experiments (identities, repetition, baseline).

Table 1. Summarizes the results of our various experiments. In the experiments, the training of the HMMs were done without the test data. We tested for invariance w.r.t. identity and w.r.t. the action. The *baseline* shows the recognition results when the test action was a single action primitives.

Leave-one-Out experiments		
Test	Noise σ	Recognition Result
Identities (Test 1)	0	0.9177
Repetitions (Test 2)	0	0.9097
Baseline (Test 3)	0	0.9417
Identities (Test 1)	0.5	0.8672
Repetitions (Test 2)	0.5	0.8710
Baseline (Test 3)	0.5	0.8649
Identities (Test 1)	1	0.3572
Repetitions (Test 2)	1	0.3395
Baseline (Test 3)	1	0.3548

All actions in the action database start and end in a resting pose. To assure that the resting pose does not effect the recognition results, we have repeated the above experiments on the action primitives where the rest poses were omitted. However, the recognition results did not change notably.

5 Conclusions

In this work we have presented an approach to recover the motion primitives from an action where the motion primitives are represented with a Hidden Markov

Model. The approach we have taken is to consider the joint distribution of the state and the action at the same time instead of using the classical maximum likelihood approach. The experiments show that the approach is able to successfully recover the action primitives in the action with a large likelihood. It is worth pointing out that in our experiments the pairwise appearance of action primitives was statistically independent. Thus, for the recovery of the action primitives no temporal constraints between the action primitives were used or exploited. Temporal constraints between the action primitives are later introduced at a higher level though action grammars.

In future work we will use a further HMM to learn sequences of action primitives from training examples to learn such an action grammar.

Acknowledgement. This work was partially funded by PACO-PLUS (IST-FP6-IP-027657).

References

1. A. Billard, Y. Epars, S. Calinon, S. Schaal, and G. Cheng. Discovering optimal imitation strategies. *Robotics and Autonomous Systems*, 47:69–77, 2004.
2. A.F. Bobick. Movements, Activity, and Action: The Role of Knowledge in the Perception of Motion. In *Royal Society Workshop on Knowledge-based Vision in Man and Machine*, London, England, February 1997.
3. H. Bourlard and N. Morgan. *Connectionist Speech Recognition: a Hybrid Approach.* Kluwer Press, 1994.
4. S. Calinon and A. Billard. Stochastic gesture production and recognition model for a humanoid robot. In *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, pages 2769–2774, IROS05, 2005.
5. S. Calinon, F. Guenter, and A. Billard. Goal-directed imitation in a humanoid robot. In *Proc. IEEE Int. Conf. on Robotics and Automation*, ICRA05, 2005.
6. N. Cuntoor and R. Chellappa. Key frame-based activity representation using antieigenvalues. In P.J. Narayanan, editor, *Proc. Asian Conference on Computer Vision*, volume 3852 of *LNCS*, pages 499–508, Hyderabad, India, January, 13-16, 2006. Springer Berlin Heidelberg.
7. B. Dariush. Human motion analysis for biomechanics and biomedicine. *Machine Vision and Applications*, 14:202–205, 2003.
8. M. Giese and T. Poggio. Neural mechanisms for the recognition of biological movements. *Nature Reviews*, 4:179–192, 2003.
9. H. Hermansky. Perceptual linear predictive (plp) analysis of speech. *Journal of Acoustical Society of America*, 87(4):1738–1725, 1990.
10. X.D. Huang, Y. Ariki, and M.A. Jack. *Hidden Markov Models for Speech Recognition.* Edinburgh University Press, 1990.
11. X.D. Huang and M.A. Jack. Semi-continous hidden markov models for speech signals. *Computer Speech and Language*, 3:239–252, 1989.
12. A.J. Ijspeert, J. Nakanishi, and S. Schaal. Movement imitation withnonlinear dynamical systems in humanoid robots. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Washington, DC, May, 2002.
13. Y. Ivanov and A. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):852–872, 2000.

14. O.C. Jenkins and M. Mataric. Deriving action and behavior primitives from human motion capture data. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Washington, DC, May, 2002.
15. O.C. Jenkins and M.J. Mataric. Deriving action and behavior primitives from human motion data. In *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, pages 2551–2556, Lausanne, Switzerland, Sept.30 – Oct.4, 2002.
16. C. Lu and N. Ferrier. Repetitive motion analysis: Segmentation and event classification. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(2):258–263, 2004.
17. H.-H Nagel. From image sequences towards conceptual descriptions. *Image and Vision Computing*, 6(2):59–74, 1988.
18. L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–15, January 1986.
19. C. Rao, A. Yilmaz, and M. Shah. View-Invariant Representation and Recognition of Actions. *Journal of Computer Vision*, 50(2), 2002.
20. L. Reng, T. Moeslund, and E. Granum. Finding motion primitives in human body gestures. In S. Gibet, N. Courty, and J.-F. Kamp, editors, *GW 2005*, pages 133–144. Springer, 2006.
21. L. Reng, T.B. Moeslund, and E. Granum. Finding motion primitives in human body gestures. In S. Gibet, N. Courty, and J.-F. Kamps, editors, *GW 2005*, number 3881 in LNAI, pages 133–144. Springer Berlin Heidelberg, 2006.
22. G. Rizzolatti, L. Fogassi, and V. Gallese. Parietal cortex: from sight to action. *Current Opinion in Neurobiology*, 7:562–567, 1997.
23. G. Rizzolatti, L. Fogassi, and V. Gallese. Neurophysiological mechanisms underlying the understanding and imitation of action. *Nature Reviews*, 2:661–670, Sept. 2001.
24. S. Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3(6):233–242, 1999.
25. A. Stolcke. An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, 21(2):165–201, 1995.
26. D.D. Vecchio, R.M. Murray, and P. Perona. Decomposition of Human Motiom into Dynamics-based Primitives with Application to Drawing Tasks. *Automatica*, 39, 2003.

Polyhedrization of Discrete Convex Volumes

Valentin E. Brimkov[1] and Reneta Barneva[2]

[1] Mathematics Department, SUNY Buffalo State College, Buffalo, NY 14222, USA
brimkove@buffalostate.edu
[2] Department of Computer Science, SUNY Fredonia, NY 14063, USA
barneva@cs.fredonia.edu

Abstract. In recent years the problem of obtaining a reversible discrete surface polyhedrization (DSP) is attracting an increasing interest within the discrete geometry community. In this paper we propose the first algorithm for obtaining a reversible polyhedrization with a *guaranteed performance*, i.e., together with a bound on the ratio of the number of facets of the obtained polyhedron and one with a minimal number of facets. The algorithm applies to the case of a *convex* DSP when a discrete surface M is determined by a convex body in $\mathbb{R}^3$. The performance estimation is based on a new lower bound (in terms of the diameter of M) on the number of 2-facets of an optimal polyhedrization. That bound easily extends to an arbitrary dimension n. We also discuss on approaches for solving the general 3D DSP.

Keywords: discrete geometry, reversible polyhedrization, polyhedron decomposition.

1 Introduction

This paper deals with a problem usually known as "discrete surface/set/volume polyhedrization" and further abbreviated DSP. In its most general form, the problem is the following. Given a set $M \subset \mathbb{Z}^3$, find a (possibly, non-convex) polyhedron P, such that the set of integer points contained in P is precisely M, i.e., $P \cap \mathbb{Z}^3 = M$. The number $f_2(P)$ of the 2-dimensional facets of P (2-facets, for short) is usually desired to be as small as possible. Often, these are required to be convex polygons; note that two adjacent polygons may be co-planar. Commonly, the set M is a discrete surface obtained from a "digitization"[1] of some (usually unknown) set $S \subset \mathbb{R}^3$ of full dimension 3. See [5,6,7,12] for related matters.

The main motivation for studying DSP problems comes from medical imaging, where discrete volumes of voxels result from scanning and MRI techniques. Since digital medical images involve a huge number of points, it is quite problematic to apply traditional rendering or texturing algorithms in order to obtain satisfactory visualization. Moreover, one can face difficulties in storing or transmitting data of that size. In medical imaging and other areas there are multiple sources of data

[1] E.g., $M = G(S)$, where $G(S)$ is the *Gauss discretization* of a set $S \subset \mathbb{R}^3$, which is defined as the set of all integer points that belong to S.

being transmitted for many diverse uses, e.g., telemedicine, tele-maintenance, mine detection, ATR, visual display, cueing, and others. In all these applications the coding compression methodology used is paramount.

To overcome such kind of difficulties, one can try to transform a discrete data set to a polyhedron P, such that the number of its 2-facets is as small as possible. The most widely used algorithm is the marching cubes method [16] (see also [13] for a recent application to the considered problem) that generates a triangulated polyhedral surface in which small triangles model local configurations of voxels. The shortcoming of this method is that the number of triangular facets in the surface may be comparable with the number of points in the original discrete object. Moreover, this method is not "reversible" in general, i.e., from the obtained polyhedrization one cannot reconstruct the original discrete object.

In recent years DSP is attracting an increasing interest in discrete geometry community and a number of papers have been devoted to it or to related matters. See [19] and the bibliography therein. Some of the proposed algorithms are reversible. Overall, they often provide satisfactory practical solutions, although sometimes the polyhedra appearance exhibits certain defects.[2] The proposed algorithms are all heuristic, usually based on certain greedy-type strategy. As a rule, these are not accompanied with an estimation of their performance. In other words, there are no results showing how far is the obtained solution from the optimal one (i.e., a polyhedron P^* for which $f_2(P^*)$ is minimal). It was recently proved that the general 3D DSP problem is strongly NP-hard [4]. The following fundamental tasks have not been addressed yet:

(A) Obtain bounds on the minimal number of polygonal facets of a polytope P that determines a given integer set M, distinguishing between the cases when the discretized set S is convex and when it is arbitrary.
(B) Design a reversible algorithm for DSP with a guaranteed performance (e.g., estimation of the ratio of the number of facets of the obtained solution and an optimal solution).

In what follows, we cope with the above tasks. In Section 2 we present a reversible polyhedrization algorithm in 3D. It works for an input set that appears to be a discrete surface. The latter is determined as the set of the "surface elements" of a discretization of a certain convex set $S \subset \mathbb{R}^3$. We also analyze the algorithm's complexity and performance (the latter in terms of "closeness" of the obtained polyhedrization to the optimal one). Our analysis is based on proving a new lower bound on $f_2(P)$ (that easily extends to an arbitrary dimension n). Thus, as a byproduct, we address Problem (A). In Section 3 we comment how the algorithm from Section 2 can be used to solve the general DSP. We discuss on possibilities of obtaining a reasonable initial polyhedrization and concern complexity issues. We conclude with some remarks and open problems in Section 4. In obtaining our results, we rely on some well-known results in the theory of lattice polytopes, integer programming, and discrete geometry.

[2] For instance, some polygons that constitute the polyhedron facets, as well as the polyhedron itself, may be non-convex, even for discrete sets that are discretizations of convex bodies (such as spheres or ellipsoids).

1.1 Some Notations

In this section we introduce some notions and notations to be used throughout the paper. Other notations will be given in the subsequent sections.

Let S be a *body* in $\mathbb{R}^3$, i.e., a subset of $\mathbb{R}^3$ of full dimension $dim(S) = 3$. By $S_{\mathbb{Z}} = S \cap \mathbb{Z}^3$ we denote its *Gauss discretization.* (Note that $S_{\mathbb{Z}}$ may be empty for a non-empty S.)

Voxel $v(p)$ is a unit grid cube centered at an integer point p. We denote the union of all voxels corresponding to points from $S_{\mathbb{Z}}$ by $Vol(S_{\mathbb{Z}}) = \cup_{p \in S_{\mathbb{Z}}} v(p)$. Clearly, $Vol(S_{\mathbb{Z}})$ is a rectilinear polytope (i.e., all its facets are parallel to the coordinate axes).

By $Surf(S_{\mathbb{Z}})$ we denote the rectilinear surface of $Vol(S_{\mathbb{Z}})$. A point $p \in S_{\mathbb{Z}}$ will be called a *surface element* of $S_{\mathbb{Z}}$ if at least one 2-facet of the corresponding voxel $v(x)$ is contained in $Surf(S_{\mathbb{Z}})$.

For a set $A \subseteq \mathbb{R}^3$, by $diam(A)$ we denote its *diameter* defined as $diam(A) = \max_{x,y \in A} ||x - y||$, where $||.||$ is the Euclidean norm. By $conv(A)$ we denote the convex hull of A. The interior and the boundary of a closed set $A \subseteq \mathbb{R}^3$ is denoted by $int(A)$ and $bd(A)$, respectively.

Given a polytope $P \subset \mathbb{R}^3$, the number of its i-facets is denoted by $f_i(P)$, $0 \le i \le 3$.

2 Reversible Polyhedrization Algorithm

In this section we present and analyze a reversible polyhedrization algorithm based on convex hull computation. It applies to *convex* DSP's for which the set S is convex.

2.1 Description of the Algorithm

We call our algorithm a *reversible convex hull computation* (RCH). The algorithm naturally has two components: Forth and Back. The former takes as an input a discrete surface M that consists of the surface elements of a certain set $S_{\mathbb{Z}}$ (where S is convex), obtained, e.g., by scan-conversion techniques.[3] Forth RCH computes the vertices and 2-facets of a convex polytope P, such that M is the set of the surface elements of $P_{\mathbb{Z}}$. Its description is straightforward.

Forth RCH (Polyhedrization): Compute the convex hull of M.

The Back RCH takes as an input a convex polytope P given by its vertex set $f_0(P)$ and its 2-facet set $f_2(P)$. It computes a set M that consists of the surface elements of $P_{\mathbb{Z}}$.

The description of Back RCH, that is more involved than Forth RCH, is given next. We start with some preliminaries.

First recall that a *discrete plane*, that appears to be a discretization of a plane $h:\ a_1x_1 + a_2x_2 + a_3x_3 + b = 0$, is a set of voxels $H(a_1, a_2, a_3, b, |a|_{\max}) =$

[3] It can also be extracted from a discrete volume $S_{\mathbb{Z}}$ by scanning each layer of it, similar to how it is done in marching cubes algorithm (see, e.g., [14], p. 301).

$\left\{(x_1, x_2, x_3) \in \mathbb{Z}^3 | 0 \leq a_1x_1 + a_2x_2 + a_3x_3 + b + \lfloor \frac{|a|_{\max}}{2} \rfloor < |a|_{\max}\right\}$, where $|a|_{\max} = \max(|a_1|, |a_2|, |a_3|)$ (see, e.g., [1,3] for basic definitions and facts). $H(a_1, a_2, a_3, b, |a|_{\max})$ is "centered" about h and provides the best approximation to it. Alternatively, we will also use the discrete plane $H'(a_1, a_2, a_3, b, |a|_{\max}) = \left\{(x_1, x_2, x_3) \in \mathbb{Z}^3 | 0 < a_1x_1 + a_2x_2 + a_3x_3 + b + \lfloor \frac{|a|_{\max}}{2} \rfloor \leq |a|_{\max}\right\}$ that has analogous properties to those of $H'(a_1, a_2, a_3, b, |a|_{\max})$. The only difference is that $H'(a_1, a_2, a_3, b, |a|_{\max})$ contains the integer points on the plane h while h' does not. We will call H the *lower* and H' the *upper* discretization of h.

Note that in the continuous case the planes $a_1x_1 + a_2x_2 + a_3x_3 + b = 0$ and $-a_1x_1 - a_2x_2 - a_3x_3 - b = 0$ coincide, while in the discrete case there might be an ambiguity. To avoid that, we impose in our definition of discrete planes the following "orientation" conditions: If $|a| = \max(|a|, |b|, |c|)$, then $a > 0$; else if $|b| = \max(|a|, |b|, |c|)$, then $b > 0$; else if $|c| = \max(|a|, |b|, |c|)$, then $c > 0$.

A discrete plane $H = H(a_1, a_2, a_3, b, |a|_{\max})$ is *functional* over a (discrete) coordinate plane, say, $H_3 = Ox_1x_2$, if for any cell $(x_1, x_2) \in H_3$ there is exactly one voxel (x_1, x_2, x_3) belonging to H. The plane H_3 is called a *functional coordinate plane* for H and denoted π_H. It is well-known (see [3]) that a discrete plane is functional over at least one of the coordinate planes $H_1 = Ox_2x_3$, $H_2 = Ox_1x_3$, or $H_3 = Ox_1x_2$. Moreover, if, e.g., $|a_3| = \max\{|a_1|, |a_2|, |a_3|\}$, then $\pi_H = H_3$. (If $\max\{|a_1|, |a_2|, |a_3|\}$ equals the absolute value of more than one of the coefficients, then the discrete plane H is functional over more than one coordinate plane.)

The Back RCH algorithm calls the following procedure that is similar to one available in more detail in [3].

Procedure A (Polygon Discretization)
Input: Space polygon $F \subset \mathbb{R}^3$ with vertices $v^{(1)}, v^{(2)}, \ldots, v^{(k)} \in \mathbb{Z}^3$, where $v^{(i)} = (v_1^{(i)}, v_2^{(i)}, v_3^{(i)})$ for $i = 1, 2, \ldots, k$.
Output: A set $I(F) \subset \mathbb{Z}^3$ that is a discretization of F.
Description:

1. Find a plane $f : a_1x_1 + a_2x_2 + a_3x_3 + b = 0$ containing F;
2. Determine a lower/upper discretization H of f;
3. Find a functional coordinate plane π_H;
4. Find the projections of $v^{(1)}, v^{(2)}, \ldots, v^{(k)}$ onto π_H (clearly, these are vertices of a convex polygon F' in π_H);
5. Compute the sides of F' and determine the set $I(F')$ of integer points that belong to the interior or the sides of F';
6. Generate the set $I(F)$ of points of H whose projections constitute the set $I(F')$.

See Fig. 1 (left and middle) for an illustration. It is not hard to realize that Procedure A takes linear time in the number of elements of $I(F)$.

After this preparation, we are able to describe Back RCH. The input to the algorithm is the set $v^{(1)}, v^{(2)}, \ldots, v^{(m)} \in \mathbb{Z}^3$ of vertices of a polytope P, and the set of its 2-facets is given as ordered lists of vertices.

Back RCH (Polytope Discretization)

(1) 1. Find a convex combination of P's vertices: $v^* = \frac{1}{m}(v^{(1)} + v^{(2)} + \ldots + v^{(m)})$, i.e., $v^* = (v_1^*, v_2^*, v_3^*)$, where $v_i^* = \frac{1}{m}(v_i^{(1)} + v_i^{(2)} + \ldots + v_i^{(m)})$ for $i = 1, 2, 3$. Clearly, $v^* \in int(P)$.

2. Compute the rank of the $(3 \times m)$-matrix $A = ((v^{(1)})^T - (v^*)^T | (v^{(2)})^T - (v^*)^T | \ldots | (v^{(m)})^T - (v^*)^T)$, where v^T denotes the transposed vector v.

(2) **(2a)** If $rank(A) = 1$, then all points from $P_{\mathbb{Z}}$ belong to a line l parallel to a coordinate axis, e.g., of the form $l : x_1 = c_1, x_2 = c_2, c_1, c_2 \in \mathbb{Z}$. We have that $conv(P_{\mathbb{Z}})$ has only two vertices $u = (c_1, c_2, x_3')$ and $v = (c_1, c_2, x_3'')$, and $M = P_{\mathbb{Z}} = \{w(c_1, c_2, x_3) \in \mathbb{Z}^3, c_1 \le x_3 \le c_2\}$.

(2b) If $rank(A) = 2$, then all points from $P_{\mathbb{Z}}$ belong to a plane p that is parallel to a coordinate plane, e.g., of the form $p : x_1 = c_1, c_1 \in \mathbb{Z}$. In other words, the points of $P_{\mathbb{Z}} = M$ belong to a convex plane polygon. These can be found in $O(|M|)$ time (see [3] for a similar procedure).

(2c) If $rank(A) = 3$, then perform the following steps:

1. For every facet $F \in f_2(P)$ do:
   Let $f : a_1x_1 + a_2x_2 + a_3x_3 + b = 0$ be the plane determined by F.
   Case 1: If $a_1v_1^* + a_2v_2^* + a_3v_3^* + b < 0$, then consider the linear constraint $f_1 : a_1x_1 + a_2x_2 + a_3x_3 + b - \lfloor \frac{|a|_{\max}}{2} \rfloor \le |a|_{\max}$.
   Case 2: If $a_1v_1^* + a_2v_2^* + a_3v_3^* + b > 0$, then consider the linear constraint $f_2 : a_1x_1 + a_2x_2 + a_3x_3 + b + \lfloor \frac{|a|_{\max}}{2} \rfloor \le |a|_{\max}$.
   (Since $v^* \in int(P)$, equality $a_1v_1^* + a_2v_2^* + a_3v_3^* + b = 0$ cannot hold.)
2. Determine the polytope P' formed by the obtained linear constraints.
3. Let F' be a facet of P'. If F' has been obtained within Case 1 (resp. Case 2), then call Procedure A to find its upper (lower) discretization.

When F' runs over all facets of P', we obtain the target set M consisting of the surface elements of $P_{\mathbb{Z}}$.

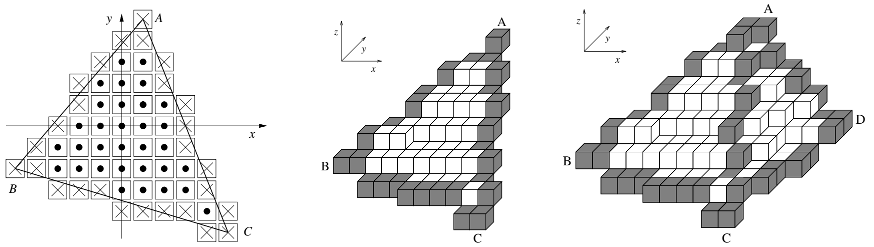


Fig. 1. *Left:* A 2D discrete triangle determined by the points $A(1, 5)$, $B(-5, -2)$, and $C(5, -5)$. The triangle border consists of the three discrete line segments determined by pairs of vertices of the continuous triangle. *Middle:* A 3D discrete triangle with vertices $A(1, 5, 4)$, $B(-5, -2, 0)$, and $C(5, -5, -2)$. Its projection onto the coordinate plane Oxy is the 2D discrete triangle on the left. The border voxels corresponding to the border pixels of the 2D discrete triangle are in dark gray. *Right:* Discrete triangular mesh with vertices $A(1, 5, 4)$, $B(-5, -2, 0)$, $C(5, -5, -2)$, $D(8, 2, 0)$.

It is not hard to realize that the described method is reversible in both directions, i.e., we have that: (i) Consecutive application of Forth and Back RCH to a set M (with the properties described above) will result into the same set M, and (ii) Consecutive application of Back and Forth RCH to a polytope P with integer vertices will result into the same polytope P. Details are left for the full-length journal version of the paper.

Remark 1. Since the set of integer points contained in a polytope P does not need to be connected [14], it is clear that the set of the surface elements of the discrete volume contained in P is not necessarily tunnel-free[4]. Conditions under which a mesh of discrete polygons (Fig. 1, right) is tunnel-free are studied in [3].

2.2 Bounds on the Number of Facets

It is not hard to realize that if in the DSP formulation S (and thus also M) is an arbitrary set, then a bound $f_i(P) = \Theta(|M|)$ holds, where $|M|$ denotes the cardinality of set M (see Fig. 2). Problem (A) becomes nontrivial if S is a convex set, that is the case we consider next.

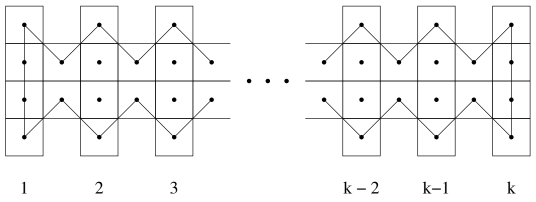


Fig. 2. Example in 2D with $|M| = 6k - 2$ and $f_i(P) = 4k - 2$

To state what we need, let $\mathcal{C}(D)$ be the family of convex bodies with $\mathcal{C}^2$ boundary and radius of curvature at every point and every direction between $1/D$ and D, where $D \geq 1$. Now let $S \in C(D)$ and $\mathbf{0} \in S$. Let $P = conv(S_{\mathbb{Z}})$. Clearly, $P_{\mathbb{Z}} = S_{\mathbb{Z}}$ and $conv(P_{\mathbb{Z}}) = conv(S_{\mathbb{Z}}) = P$. Let P^* be a convex polytope with a minimal number of 2-facets, such that $P^*_{\mathbb{Z}} = P_{\mathbb{Z}} = S_{\mathbb{Z}}$. Then, $conv(P^*_{\mathbb{Z}}) = P$. Denote $d_P = diam(P)$, $d_{P^*} = diam(P^*)$, and $d_S = diam(S)$. Clearly, $diam(P) \leq diam(P^*)$ and $diam(P) \leq diam(S)$. It is easy to see that any of the relations $diam(P^*) < diam(S)$, $diam(P^*) = diam(S)$, and $diam(P^*) > diam(S)$ is possible. In what follows, we will suppose that these diameters are sufficiently large. We will show that there exists a constant α, such that $\frac{f_2(P)}{f_2(P^*)} \leq \alpha.\log^2(1 + d_{P^*})$, i.e., the number of facets of the solution provided by Forth RCH algorithm is within a factor $\alpha. \log^2(1 + d_{P^*})$ from the optimal solution. In the proof we will use the following well-known facts.

[4] Here we assume the common notions of a tunnel and tunnel-freedom, known from the classical topology, applied to the polyhedron $Vol(S_{\mathbb{Z}})$.

Lemma 1. *Let P and P' be bounded polytopes with vertices $v_1, v_2, \ldots, v_k$ and $v'_1, v'_2, \ldots, v'_k$, respectively. If $||v_i - v'_i|| < \varepsilon$, where ε is a sufficiently small positive real number, then P and P' have the same number of k-facets $(0 \leq k \leq 2)$.*

This last fact, that can be classified as belonging to the mathematical folklore, easily follows from theory of polyhedra (see, e.g., [17]). It implies that, given a polytope P, there is a rational polytope P', to which the conditions of Lemma 1 apply. Thus in what follows, one can assume that both convex polytopes P and P^* are rational.

Lemma 2. [2] *Let $K \in \mathcal{C}(D)$ and $\bar{K} = conv(K \cap \mathbb{Z}^n)$. Then for every $n \geq 2$ there are positive constants $c_1(n)$ and $c_2(n)$ depending only on n, such that for all $k \in \{0, 1, \ldots, n-1\}$, $c_1(n)d^{n\frac{n-1}{n+1}} \leq f_k(\bar{K}) \leq c_2(n)d^{n\frac{n-1}{n+1}}$, where $d = diam(K)$ is sufficiently large. In particular, for $n = 3$, we have*

$$c_1 d^{3/2} \leq f_k(\bar{K}) \leq c_2 d^{3/2}, \tag{1}$$

for some positive constants c_1 and c_2.

Lemma 3. [8] *Let $Q = \{x \in \mathbb{R}^n : Ax \leq \mathbf{b}, A \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{Z}^m\}$. Consider the polytope $conv(Q_{\mathbb{Z}})$ and denote by $V(A, \mathbf{b})$ the set of its vertices. Then $V(A, \mathbf{b}) \leq c(n)m^{\lfloor n/2 \rfloor} \log^{n-1}(1 + d)$, where $d = \max_{x,y \in Q} \max_{j=1,2,\ldots,n} |x_j - y_j|$ and $c(n)$ is a positive constant depending only on n. In particular, for $n = 3$, we have*

$$V(A, \mathbf{b}) \leq c.m \log^2(1 + d). \tag{2}$$

for some positive constant c.

Clearly, Lemma 3 holds also if d is defined by the Euclidean metric.

Estimation of Forth RCH performance is implied by the following theorem that provides a lower bound on the number of facets of P^*.

Theorem 1. *There is an absolute constant c, such that*

$$f_2(P^*) \geq c \frac{d_S^{3/2}}{\log^2(1 + d_{P^*})} \geq c \frac{d_P^{3/2}}{\log^2(1 + d_{P^*})}.$$

Proof. From (1) we have that there exist $c_1, c_2 > 0$ such that $c_1 d^{3/2} \leq f_k(P) \leq c_2 d^{3/2}$. Denote for short $m = f_2(P)$, $m_* = f_2(P^*)$, $v = f_0(P)$, and $v_* = f_0(conv(P^*_{\mathbb{Z}}))$. Clearly, $v = v_*$. We have

$$m^* \leq m \leq c_2 d_S^{3/2}. \tag{3}$$

Multiplying by $\frac{c_1}{c_2}$ both sides of the last inequality in (3) and using once again Lemma 2, we obtain $\frac{c_1}{c_2} m \leq c_1 d_S^{3/2} \leq v$. Now, keeping in mind Lemma 1, from (2) we obtain $v = v^* \leq c' m_* \log^2(1 + d_{P^*})$ and thus $c_1 d_S^{3/2} \leq c' m_* \log^2(1 + d_{P^*})$, where c' is a positive constant. From here we obtain consecutively

$$m_* \geq c'' \frac{d_S^{3/2}}{\log^2(1 + d_{P^*})},$$

and

$$m_* = f_2(P^*) \geq c \frac{d_S^{3/2}}{\log^2(1+d_{P^*})} \geq c \frac{d_P^{3/2}}{\log^2(1+d_{P^*})},$$

for some positive constants c'' and c, respectively. □

From Theorem 1 we immediately obtain the following corollaries.

Corollary 1. *There is a constant $\alpha_2 \in \mathbb{Z}_+$, such that $\frac{f_2(P)}{f_2(P^*)} \leq \alpha_2 \log^2(1+d_{P^*})$.*

If S is a sphere, then there are constants $\lambda_1, \lambda_2 \in \mathbb{R}_+$ such that $d = diam(S) = \lambda_1\ diam(P) = \lambda_2\ diam(P^*)$. Then we obtain

Corollary 2. *Let S be a sphere in $\mathbb{R}^3$ with a radius r. Then there are constants $\beta_2, \beta_2' \in \mathbb{Z}_+$, such that $f_2(P^*) \geq \beta_2 \frac{r^{3/2}}{\log(r+1)}$ and $\frac{f_2(P)}{f_2(P^*)} \leq \beta_2' \log^2(r+1)$.*

Remark 2. It is not hard to realize that the result of Theorem 1 easily generalizes to higher dimensions. More precisely, one can show that for every $n \geq 2$ there is a constant $c(n)$ depending only on n such that

$$f_{n-1}(P^*) \geq c(n) \frac{d_S^{\frac{n(n-1)}{(n+1)\lfloor n/2 \rfloor}}}{\log^{\frac{n-1}{\lfloor n/2 \rfloor}}(1+d_{P^*})} \geq c(n) \frac{d_P^{\frac{n(n-1)}{(n+1)\lfloor n/2 \rfloor}}}{\log^{\frac{n-1}{\lfloor n/2 \rfloor}}(1+d_{P^*})}. \tag{4}$$

For $n = 3$, (4) provides the result of Theorem 1 and its corollaries. In the case when S is a disc in $\mathbb{R}^2$ with a radius r, (4) implies that there are constants $\beta_1, \beta_1' \in \mathbb{Z}_+$, such that $f_1(P^*) \geq \beta_1 \frac{r^{2/3}}{\log(r+1)}$ and $\frac{f_1(P)}{f_1(P^*)} \leq \beta_1' \log(r+1)$. These last two inequalities have been recently obtained in [9].

2.3 Complexity

The complexity of Forth RCH matches the complexity of convex hull computation. It is well-known that in arbitrary dimension n finding the convex hull of m points of $\mathbb{R}^n$ takes at worst $O(m^{\lfloor \frac{n}{2} \rfloor + 1}) + O(m^{\lfloor \frac{n}{2} \rfloor} \log m)$ time [18]. Then, in dimension three, the complexity of Forth RCH is $O(|M|^2)$. Since Back RCH involves convex hull computation, its time complexity is analogous to the one of Forth RCH.

3 Towards Solution of the General DSP

In this section we propose an approach for solving DSP in its general form when S is an arbitrary body in $\mathbb{R}^3$. The algorithm consists of three phases:

1. Obtaining an initial polyhedrization of M, that, in general, will be a non-convex polyhedron Q.
2. Partitioning Q into convex polytopes.

3. Applying the algorithm from Section 2 to each of the convex polytopes obtained in Phase 2.

Phase 1: Obtaining an Initial Polyhedrization. The objective here is to obtain a polyhedron with a comparatively small number of 2-facets, which can be decomposed into a comparatively small number of convex polytopes.

One possible approach is to use the well-known marching cubes algorithm [16]. However, a polyhedron obtained this way has, as a rule, a great number of small 2-facets. Moreover, in general, this algorithm is not reversible and the obtained polyhedrization may not be hole-free (see discussion in [19] as well as pp. 301-303 of [14]).

Another starting point may be the rectilinear polyhedron $Vol(S_{\mathbb{Z}})$. However, in practice such polyhedra feature a great number of *notches* (locations causing non-convexity[5]). This may result in a very large number of "small" convex polytopes in the convex decomposition of Phase 2. Note that in dimension three, the optimization decomposition problem is NP-complete [10].

Another possible approach follows from [19]: Apply the algorithm from [19] to partition the discrete surface into discrete polygonal patches, then for each discrete patch construct the corresponding (possibly non-convex) polygon. The second step of the above procedure is not readily available in [19], so its realization is seen as a further task.

Phase 2: Polyhedron Decomposition. Given a non-convex polyhedron obtained in Phase 1, our task is to decompose it into an as small as possible number of convex polytopes. It has been shown that the optimization version of this problem is strongly NP-hard [15]. Therefore, it makes sense to look for an approximation solution to it. For this, one can use, e.g., the algorithm of Hershberger and Snoeyink [11] that provides decomposition into $O(r^2)$ convex polytopes in $O(nr + r^{\frac{7}{3}})$ time, where n and r are the number of edges and the number of notches of Q, respectively.

4 Concluding Remarks

In this paper we proposed an algorithm for obtaining a reversible discrete surface polyhedrization for a set of voxels contained in a convex body. We also provided time complexity bounds for the algorithm as well as a bound on its performance in terms of the diameter of M. An open question is whether the obtained performance bound is tight. Obtaining similar bounds for the case of an arbitrary discrete volume is another challenging task.

Acknowledgements

The authors thank the three anonymous referees for some useful remarks.

[5] Notch (or *reflex edge*) is an edge of a polyhedron where the inner dihedral angle subtended by two incident facets is greater than 180 degrees.

References

1. Andres, E., R. Acharya, C. Sibata, Discrete analytical hyperplanes, *Graphical Models and Image Processing* **59** (1997) 302–309
2. Bárány, I., D.G. Larman, The convex hull of the integer points in a large ball, *Math. Ann.* **312** (1998) 167–181
3. Barneva, R.P, V.E. Brimkov, Ph. Nehlig, Thin discrete triangular meshes, *Theoretical Computer Science* **246** (1-2) (2000) 73–105
4. Brimkov, V.E., Discrete volume polyhedrization is strongly NP-hard, CITR-TR-179, Centre for Image Technology and Robotics, University of Auckland, New Zealand, March 2006, http://citr.auckland.ac.nz/techreports/show.php?id=179
5. Brimkov, V.E., E. Andres, R.P. Barneva, Object discretizations in higher dimensions, *Pattern Recognition Letters* **23** (2002) 623–636
6. Brimkov, V.E., D. Coeurjolly, R. Klette, Digital Planarity - A Review, CITR-TR 142, 2004
7. Brimkov, V.E., R. Klette, Curves, hypersurfaces, and good pairs of adjacency relations, LNCS, No 3322, Springer Verlag (2004) 270–284
8. Chirkov, A.Y., On the relation of upper bounds on the number of vertices of the convex hull of the integer points of a polyhedron with its metric characteristics (in Russian), In: Alekseev, V.E. et al. (Eds), *Proc. 2nd Internat. Conf. "Mathematical Algorithms,"* N. Novgorod, Nizhegorod University Press, 1997, pp. 169–174
9. De Vieilleville, F., J.-O. Lachaud, F. Feschet, Maximal digital straight segments and convergence of discrete geometric estimators, Proc. *SCIA'05* (2005) 988–997
10. Dielissen, V.J., A. Kaldewaij, Rectangular partition is polynomial in two dimensions but NP-complete in three, *Information Processing Letters* **38** (1991) 1–6
11. Hershberger, J.E., J.S. Snoeyink, Erased arrangements of lines and convex decompositions of polyhedra, *Computational Geometry: Theory and Applications* **9** (1998) 129–143
12. Kaufman, A., D. Cohen, R. Yagel, Volume graphics, *IEEE Computer* **26**(7) (1993) 51–64
13. Kenmochi, Y., A. Imiya, N.F. Ezquerra, Polyhedra generation from lattice points, In: S. Miguet, A. Montanvert, and A. Ubéda (Editors), *Discrete Geometry for Computer Geometry*, LNCS 1176 (2000) 127–138
14. Klette, R., A. Rosenfeld, *Digital Geometry - Geometric Methods for Digital Picture Analysis*, Morgan Kaufmann, San Francisco, 2004
15. Lingas, A., The power of non-rectilinear holes, In: *Proc. 9th Internat. Colloc. on Automata, Languages, and Programming*, LNCS 140 (1982) 369–383
16. Lorensen, W.E., H.E. Cline, Marching cubes: a high resolution 3d surface construction algorithm, *Computer Graphics* **21**(4) (1987) 163–169
17. Papadimitriou, Ch., K. Steiglitz, *Combinatorial Optimization*, Prentice-Hall, New Jersey, 1982
18. Preparata, F., M.I. Shamos, *Computational Geometry: An Introduction*, Springer, New York, 1985
19. Sivignon, I., F. Dupont, J.-M. Chassery, Decomposition of three-dimensional discrete objects surface into discrete plane pieces, *Algorithmica* **38** (2004) 25–43

Automatic Camera Calibration and Scene Reconstruction with Scale-Invariant Features

Jun Liu and Roger Hubbold

School of Computer Science, University of Manchester
Manchester, M13 9PL, United Kingdom
{jun, roger}@cs.man.ac.uk

Abstract. The goal of our research is to robustly reconstruct general 3D scenes from 2D images, with application to automatic model generation in computer graphics and virtual reality. In this paper we aim at producing relatively dense and well-distributed 3D points which can subsequently be used to reconstruct the scene structure. We present novel camera calibration and scene reconstruction using scale-invariant feature points. A generic high-dimensional vector matching scheme is proposed to enhance the efficiency and reduce the computational cost while finding feature correspondences. A framework for structure and motion is also presented that better exploits the advantages of scale-invariant features. In this approach we solve the "phantom points" problem and this greatly reduces the possibility of error propagation. The whole process requires no information other than the input images. The results illustrate that our system is capable of producing accurate scene structure and realistic 3D models within a few minutes.

1 Introduction

The possibility of being able to acquire 3D information from 2D images has attracted considerable attention in recent years. It offers promising applications in such areas as archaeological conservation, scene-of-crime analysis, architectural design, movie post-processing, to name but a few. The idea of automatic reconstruction from images is intriguing because, unlike other techniques which usually require special devices to obtain the data (e.g. laser scanner, ultrasound), the digital images are readily available. Although reconstructing 3D models from 2D images is a very difficult problem, recent years have seen several theoretical breakthroughs, and a few systems have already been built. However, most of the systems only work under restrictive conditions, and are not readily applicable to more general cases.

One of the most important stages of scene reconstruction is structure from motion (SfM), which determines the camera poses and scene structure based on image information alone. Feature points are first extracted from each input image, and they are tracked to provide information about the relations between the images. Therefore, feature extraction and tracking act as a starting point for scene reconstruction, and their performance largely determines the overall reliability of the reconstruction algorithm. Two of the most popular feature tracking algorithms are the Harris corner detector [1] followed by Sum of Squared Difference (SSD) matching [2, 3, 4, 5], and the Kanade-Lucas-Tomasi (KLT) tracker [6, 7]. These algorithms work well when the baseline (i.e. viewpoint change between images) is relatively small and the appearance of the features doesn't change much across subsequences. However, this condition does not hold when the input data is a sequence of "sparse" images instead of a "dense" video stream, or where the appearances of the features change significantly with respect to the viewpoint. Therefore, a more robust feature tracking method is desirable to form a good foundation for the scene reconstruction problem.

The Scale Invariant Feature Transformation (SIFT), first proposed by Lowe[8, 9], extracts distinctive features which act as descriptors of local image patches. These features are largely invariant to image scale and rotation, and partially invariant (i.e. robust) to affine distortion, change in 3D viewpoint, addition of noise and change in illumination. SIFT has become well-accepted by the computer vision community. A recent evaluation by Mikolajczyk and Schmid [10] suggested that the SIFT-based descriptors performed the best among many other local descriptors, in terms of distinctiveness and robustness to various changes in viewing conditions. Successful applications of the SIFT algorithm have been reported in the areas of object recognition [8, 9], panorama stitching [11] and augmented reality [12].

Due to the invariant properties of SIFT, it can potentially tackle the problem of wide baseline matching and matching between significantly changing features. There is, however, very little literature about the application of SIFT in such areas as camera calibration and scene reconstruction. A similar but different work to ours is by Gordon and Lowe [12], where SIFT features are extracted and matched to relate any two images from an image sequence. In this paper we propose a more complete algorithm for SIFT feature matching and SfM computation. Our system is different from others in that we not only use SIFT features for camera pose estimation, but also their reconstructed 3D positions for scene analysis.

The rest of the paper is organised as follows: Section 2 introduces a new feature matching algorithm based on SIFT, which improves the efficiency without compromising its accuracy. Section 3 discusses a novel framework for SfM, with the advantage that it can match features from non-adjacent images, thus solving the problem of "phantom points" and making the system less prone to error propagation. Section 4 shows some experimental results to validate our method and Section 5 concludes our work.

2 A New Approach for SIFT Feature Matching

2.1 Related Work

The SIFT algorithm computes, for each keypoint, its location in the image as well as a distinctive 128-dimension descriptor vector associated with it. Matching a keypoint to a database of keypoints is usually done by identifying its nearest neighbour in that database . The nearest neighbour is defined as the keypoint with minimum Euclidean distance to the descriptor vector. To reduce the number of spurious matches, the ratio R of the distance of the closest neighbour D to that of the second closest neighbour D′ is computed. The matches with a ratio greater than a certain threshold (0.8 is suggested by Lowe) are discarded.

Due to the high dimensionality of the keypoints, the matching process is relatively expensive. The naive exhaustive search has a complexity of $\mathcal{O}(nmd)$ where n is the number of keypoints in the database, m is the number of keypoints to be matched, and d is the dimension of the descriptor vector. The best algorithms, such as a k-d tree, provide no speedup over exhaustive search for more than about 10 dimensional spaces [9]. Therefore, two approximate matching algorithms have been proposed, namely Best-Bin-First(BBF) [13] and PCA-SIFT [14]. The BBF algorithm is very similar to the k-d tree algorithm, except that the BBF algorithm restricts the search step so that it sets an absolute time limit on the search. As a result, the BBF algorithm returns a nearest neighbour at a high probability. However, our experiment shows that as the number of keypoints and the dimension increases, the BBF algorithm provides no significant speedup over the standard matching method. PCA-SIFT, on the other hand, reduces the dimensionality based on Principal Component Analysis. Both algorithms incur a certain amount of loss of correct matches. In our system SIFT is applied to act as a starting point for structure & motion and camera calibration, so it is desirable that the data is as noise-free as possible.

2.2 Problem Specification

A formal specification of the matching problem is first outlined. Suppose we have in the database n points $\mathbf{P} = \{\mathrm{P}_0, \mathrm{P}_1, ..., \mathrm{P}_{n-1}\}$, each of which comprises of a d dimensional descriptor vector $[V_0, V_1, ..., V_{d-1}]$. We want to match m points $\mathbf{P}' = \{\mathrm{P}'_0, \mathrm{P}'_1, ...\mathrm{P}'_{m-1}\}$, each with a descriptor vector of the same dimension $[V'_0, V'_1, ..., V'_{d-1}]$, to the database, in order to obtain a set of matched pairs $\mathbf{S} = \{(\mathrm{P}, \mathrm{P}') \mid \mathrm{P} \leftrightarrow \mathrm{P}', \mathrm{P} \in \mathbf{P}, \mathrm{P}' \in \mathbf{P}'\}$. A matched pair $(\mathrm{P}_i, \mathrm{P}'_j)$ has the property that P_i is the nearest neighbour of P'_j in the database $\mathbf{P}$, i.e. ,

$$\forall \mathrm{P}_k \in \mathbf{P} \qquad \mathrm{D}(\mathrm{P}_i, \mathrm{P}'_j) \leq \mathrm{D}(\mathrm{P}_k, \mathrm{P}'_j) \tag{1}$$

where $\mathrm{D}(\mathrm{P}_i, \mathrm{P}'_j)$ is the Euclidean distance between the two descriptor vectors associated with the two keypoints. Furthermore, if P_k is the second nearest neighbour to P'_j in the database, then another constraint should be satisfied that

$$\mathrm{D}(\mathrm{P}_i, \mathrm{P}'_j) \ / \ \mathrm{D}(\mathrm{P}_k, \mathrm{P}'_j) \leq thr \tag{2}$$

where thr is the threshold value, normally 0.8. This thresholding is designed to make sure that the match is distinctive enough from other possible matches, so as to discard many spurious matches.

2.3 The Algorithm

Here we present a new method which improves the efficiency without compromising its accuracy. Our algorithm first performs a Principal Component Analysis

(PCA) on the two sets of keypoints $\mathbf{P}$ and $\mathbf{P}'$, or more specifically, on the descriptor vectors associated with them. PCA is essentially a multivariate procedure which rotates the data such that maximum variabilities are projected onto the axes. In our case, the descriptor vector sets

$$\mathbf{V} = \{(V_0^i, V_1^i, ...V_{d-1}^i) \mid \mathrm{P}_i \in \mathbf{P}\} \tag{3}$$

$$\mathbf{V}' = \{({V'}_0^j, {V'}_1^j, ...{V'}_{d-1}^j) \mid \mathrm{P}'_j \in \mathbf{P}'\} \tag{4}$$

are transformed into

$$\widehat{\mathbf{V}} = \{(\widehat{V}_0^i, \widehat{V}_1^i, ...\widehat{V}_{d-1}^i) \mid \mathrm{P}_i \in \mathbf{P}\} \tag{5}$$

$$\widehat{\mathbf{V}'} = \{(\widehat{V'}_0^j, \widehat{V'}_1^j, ...\widehat{V'}_{d-1}^j) \mid \mathrm{P}'_j \in \mathbf{P}'\} \tag{6}$$

with $\widehat{V}_0$ and $\widehat{V'}_0$ representing the dimension of the greatest amount of variation, $\widehat{V}_1$ and $\widehat{V'}_1$ representing the dimension of the second greatest amount of variation, and so on.

The next step is that for every keypoint P'_j in $\mathbf{P}'$, two initial full Euclidean distances between P'_j and the first two elements in $\mathbf{P}$, P_0 and P_1, are computed. These initial distances, $\mathrm{D}(\mathrm{P}'_j, \mathrm{P}_0)$ and $\mathrm{D}(\mathrm{P}'_j, \mathrm{P}_1)$, are compared, with the smaller one assigned to the nearest distance Nd, and the bigger one assigned to second nearest distance Snd.

After the initialisation, the comparison continues, but without the necessity to compute the full Euclidean distance for each keypoint in $\mathbf{P}$. Suppose now we want to test the keypoint P_i and see whether it is a nearer neighbour to P'_j than the current nearest one. We start by computing the difference of the vector in the first dimension $\mathrm{D}^2 \leftarrow (\widehat{V'}_0^j - \widehat{V}_0^i)^2$, and compare it with the nearest distance squared Nd^2. If $\mathrm{D}^2 \geq Nd^2$, which indicates that P_i cannot become the nearer neighbour, then it is unnecessary to compute the rest of the dimensions. If $\mathrm{D}^2 < Nd^2$, then the process continues by adding the difference of the vector in the second dimension, $\mathrm{D}^2 \leftarrow \mathrm{D}^2 + (\widehat{V'}_1^j - \widehat{V}_1^i)^2$. The aim of this method is to avoid unnecessary computations in the dimensional space, i.e., to more quickly discard any keypoint which is unlikely to be the nearest neighbour. If, after going though all the dimensions d, $\mathrm{D}^2 < Nd^2$ still holds, then we identify P_i as the new nearest neighbour by assigning Nd to Snd, and assigning D to Nd.

The process continues until it reaches the end of the list P_{n-1}. The final stage is to compute the ratio R of the distance of the nearest neighbour Nd to that of the second nearest neighbour Snd: $R = Nd/Snd$. If R is below a certain threshold thr, then the matched pair is added to the set $\mathbf{S}$, otherwise there is no reliable match. The role PCA plays here is to re-arrange the order of the dimensions so that the ones with larger variation come before the ones with smaller variation. This allows this algorithm to execute more quickly, i.e. to discard faster the keypoint which is not the nearest neighbour.

2.4 Experimental Results

We use an experimental setup where we match 2 sets of keypoints of same size. This assumption is valid when the SIFT algorithm is applied to camera calibration, in which case the number of keypoints detected for each frame does not vary very much. The number of keypoints is in a range from 250 to 4000. We use Pollefeys' Castle sequence [15] to illustrate the algorithm. Two images are randomly selected from the Castle Sequence, from which the SIFT algorithm can detect up to around 4200 keypoints for each image. Then a random subset of the detected keypoints is selected and matched using the standard exhaustive search algorithm as well as our new algorithm. In this experiment 8 different image pairs are tested, and each pair is matched 3 times with a certain number of keypoints selected. The average time spent on keypoint matching is compared and the result is shown in Figure 1. The result suggests that for the cases where

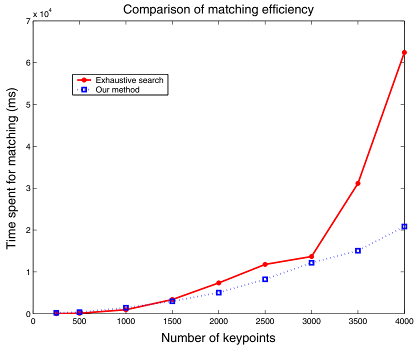


Fig. 1. Efficiency comparison between the standard exhaustive search algorithm and our improved algorithm

the number of keypoints is less than 1000, the performance of our algorithm is only slightly worse than the standard algorithm. This is because the PCA computation in our algorithm introduces an overhead, which offsets the speedup for modest point sets. However, our algorithm significantly outperforms the original one when the number of keypoints exceeds 3000. Therefore, our algorithm is ideal for a large keypoint database, as it greatly improves the efficiency while preserving the accuracy.

3 A Novel Framework for Structure from Motion

3.1 The "Phantom Points" Problem

The classical SfM only relates an image to the previous one. It is implicitly assumed that once a point is out of frame or occluded, it will not reappear. Although this is true for many sequences, the assumption does not always hold. Imagine a certain point becomes occluded for several frames in the middle of the sequence, but becomes visible again for the rest of the sequence. The classical SfM method will generate two different 3D points although they are supposed to be the same 3D point. Here we coin the term *phantom points*, referring to points which the algorithm generates, but which do not really exist separately. The "phantom points" problem has so far not been properly addressed in the computer vision literature. Unfortunately, the problem often occurs in real image sequences, where there are foreground occlusions, or the camera moves back and forth.

3.2 A New Framework for SfM

We start by extracting the keypoints from the first image and inserting them into a list of vectors. Each keypoint P has three properties associated with it: its location in the image, `coord`, its feature vector, `fvec`, and the frame number of the image which it is from, `fnum`. After inserting the keypoints of the first image, the list is illustrated in Figure 2(a) (with `fnum` marked):

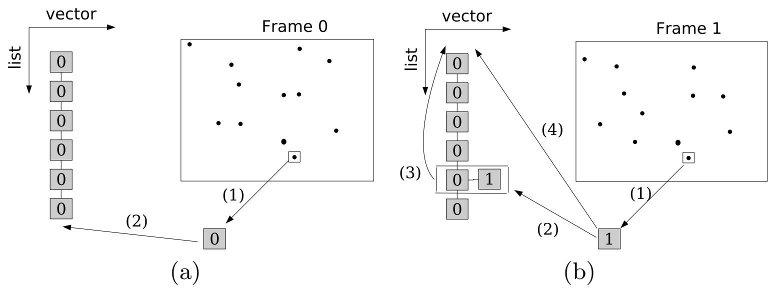


Fig. 2. (a): Adding the first frame: (1) Features are extracted with SIFT, each of which contains the information of its location in the image `coord`, its feature vector `fvec`, and the frame number of the image which it is from `fnum`. Here only `fnum` is illustrated; (2) The keypoints are inserted at the back of the list. (b): Adding the second frame: (1) Features are extracted, which are matched to the list; (2) For those which find a match, we extend the vector and move the pair (3) to the front of the list; (4) For those which cannot find a match, we insert them at the front of the list.

The keypoints from the second image are extracted and matched with the method described in Section 2.3. For those which do not match any keypoints in Frame 0, we simply insert them at the front of the list. For those which do match, we extend the vector and move the pair to the front of the list, which is illustrated in Figure 2(b).

From the matched pairs a fundamental matrix `F` is computed. Based on `F`, spurious matches (the ones which do not adhere to the epipolar constraint) are detected. The false matches are, however, not removed from the list as in traditional methods. Instead, false matches are split: we remove the last item from the vector and insert it at the front of the list (See Figure 3(a)). This way the keypoints that SIFT detects are utilised to the maximum: the falsely matched keypoints are given another chance to match the keypoints from later frames.

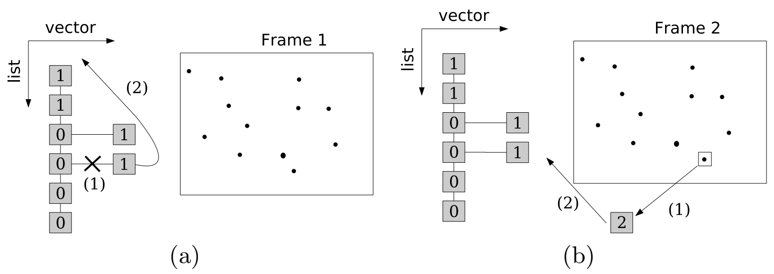


Fig. 3. (a): Rejecting outliers: outliers are detected based on the computed fundamental matrix F. If a pair is detected as an outlier, then the algorithm (1) removes the unmatched features and (2) insert it at the front of the list. (b) Adding the third frame: (1) Features are extracted, and (2) matched to the last item of each vector. Note that the keypoints from Frame 2 can be matched to both those from Frame 1 and those from Frame 0.

The initial poses and structure are computed the same way as the traditional method. When a new view is available, the extracted keypoints are compared to the last item of each vector in the list. Again the outliers are "discarded" by splitting the matches rather than removing them. Figure 3(b) shows, as an example, the status of the list after adding Frame 2. Note that the keypoints from Frame 2 can be matched to both those from Frame 1 and those from Frame 0. This is important because the matching is no longer restricted to adjacent frames. The framework described here natively solves the "phantom points" problem.

4 Experimental Results

Our SfM framework has been tested with the dinosaur sequence [16] (see Figure 5(b)) from the Robotics Group, University of Oxford. Our work is different from theirs [16] in that we do not require any prior knowledge of the input sequence, i.e. we do not need to know whether it is a turntable sequence or not. To provide a comparison, we first reconstruct the sequence with the traditional

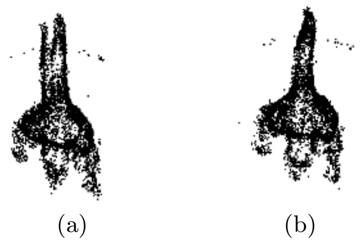


Fig. 4. Comparison of the reconstruction with the traditional method and our method (view from the top). (a): With the traditional method, severe "phantom points" lead to misalignment of the tail. (b): There are no "phantom points" with our method; thus the shape of the tail is consistent.

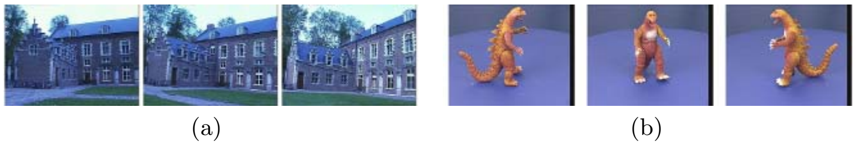


Fig. 5. Image sequences used in the comparison of the reprojection error. (a) Castle sequence, 3 samples of 28 images; (b) Dinosaur sequence, 3 samples of 37 images.

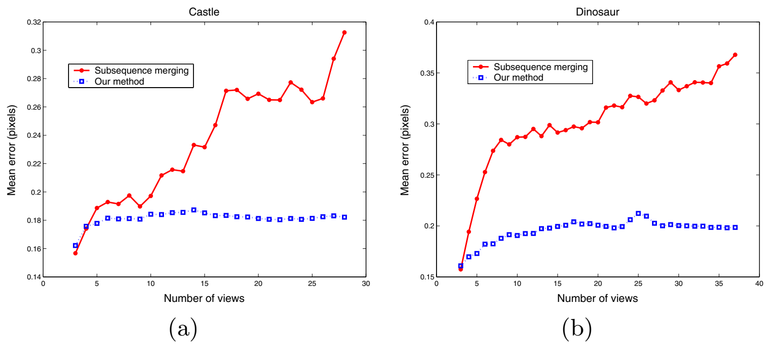


Fig. 6. Comparison of mean reprojection error between subsequence merging and our method: (a) Castle sequence and (b) Dinosaur sequence

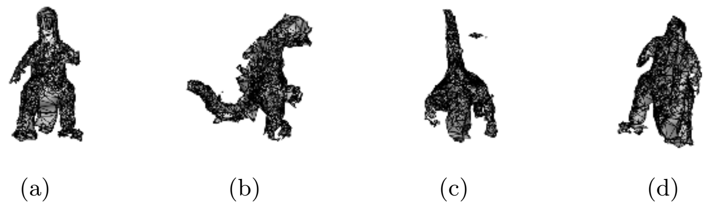


Fig. 7. Meshed model of the Dinosaur sequence: (a)front view, (b)side view, (c)top view and (d)back view. The model comprises of more than 6000 robustly tracked and well-distributed 3D points. With our system the whole reconstruction process (from the initial input images to the final output model) requires less than 10 minutes.

method, where the features from current frame only relate to the previous adjacent frame. To better illustrate the reconstruction of feature points, we generate a novel view from the top of the dinosaur. From Figure 4(a) it is obvious that this method suffers from the "phantom points" problem: the tail of the dinosaur exhibits slight misalignment, although the dinosaur is supposed to have only one integral tail. Note that the misalignment effect is exaggerated by error propagation in camera auto-calibration. The sequence is then tested with our new SfM framework, where features from current frame are matched to those from all the previous frames, and the result is shown in Figure 4(b).

Quantitative assessment was carried out to validate the advantages of our proposed SfM framework. Two publicly available image sequences were chosen:

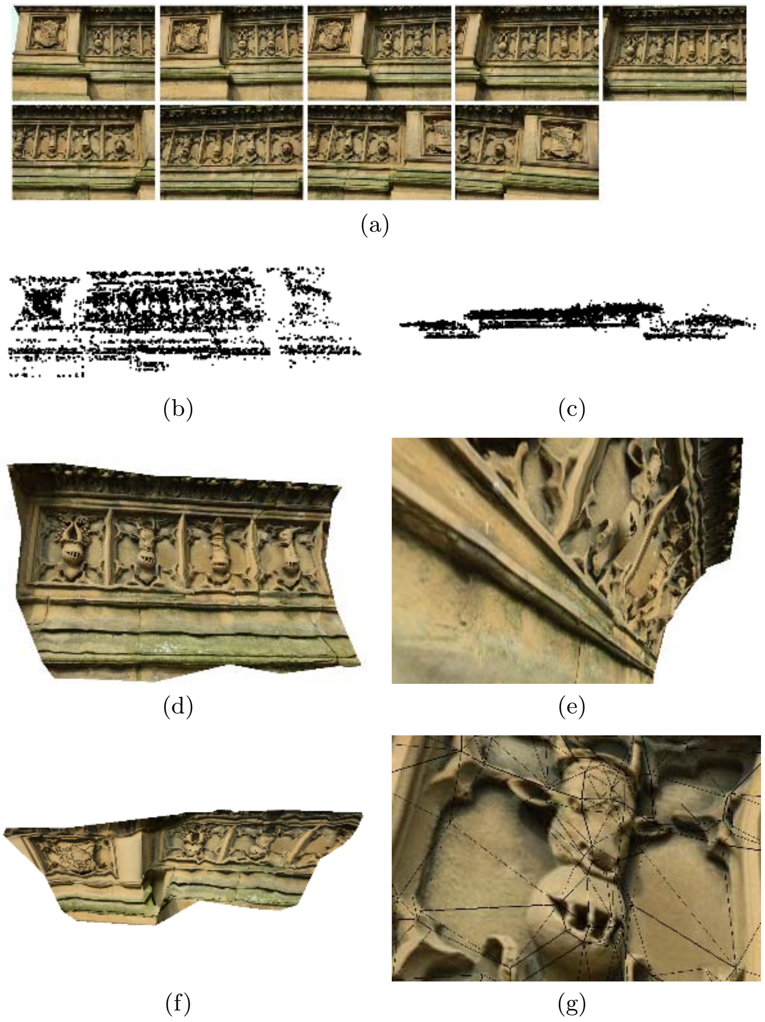


Fig. 8. (a) A challenging test case consisting of 9 images, each of which is 1024 × 679 in resolution. This small set of images involve complicated camera movements including scaling and wide baseline translation. The surface point reconstruction viewed (b) from the front and (c) from the top illustrates that our system performs well in linking the widely separated frames into a consistent scene structure. In (d) a reference image is selected and a textured model is viewed from the front. We move the viewpoint to somewhere very different from the original ones and a novel view is shown in (e) from the very left and (f) from the top (textured with a different reference image). The straightness of lines demonstrates the accurate recovery of the depth information. We further analyse the quality of reconstruction by super-imposing the triangular meshes onto the textured model. The zoomed-in details are shown in (g). Meshes are more refined in complicated areas than in plain areas. This is desirable because the computational resources are better distributed, biasing towards fine recognisable details in both scene reconstruction and model rendering. The reconstruction finishes within 5 minutes on a 2GHz processor.

the Castle sequence [15] (see Figure 5(a)) and the Dinosaur sequence [16] (see Figure 5(b)). A commonly used criterion to analyse the quality of reconstruction is the "reprojection error", which is the geometric Euclidean distance (or L_2 norm) between the projection of the reconstructed 3D point and the measured image point. In our experiments the mean reprojection error for all the reconstructed 3D points is used as an indication for the quality of the SfM methods. Our results are compared to the results using subsequence merging [17, 18].

Even though the subsequence merging technique performs well in constraining the overall mean reprojection error, it still shows moderate signs of error propagation. Results in Figures 6(a) and 6(b) suggest that our method is significantly less prone to error propagation compared to the subsequence merging technique. It is also interesting to see that our method performs surprisingly well for the Dinosaur sequence, considering the fact that it is a turntable sequence involving frequent self-occlusions. The ability to relate non-adjacent frames is important for pose estimation, as it results in projection matrices in a more consistent projective framework.

Figure 7 shows the reconstructed model of the Dinosaur sequence. Our system recovers 6105 surface points which are subsequently meshed using Cocone[19]. Several views are taken from positions very different from the original viewpoints and the results indicate that the model structure is accurately reconstructed. The whole reconstruction process requires no user intervention and finishes within 10 minutes on a 2GHz processor. Our system was further tested with photos taken with a Nikon D70s digital camera. Figure 8 shows our reconstruction of a sculptured memorial.

5 Conclusions and Future Work

We have presented a scene reconstruction system based on scale-invariant feature points. Our system is carefully designed such that the features from non-adjacent frames can be matched efficiently. We solve the "phantom points" problem and greatly reduce the chance of error propagation. Experimental results show that relatively dense and well-distributed surface points can be recovered. Our system assigns refined and detailed meshes to complicated areas, but coarse and simple meshes to plain areas. This is desirable because the computational resources are better distributed, biasing towards fine recognisable details in both scene reconstruction and model rendering.

Future work includes a more sophisticated meshing scheme and inclusion of edge information to better represent the scene structure.

References

[1] Harris, C.J., Stephens, M.: A combined corner and edge detector. In: Proceedings of 4th Alvey Vision Conference. (1988) 147–151

[2] Zhang, Z., Deriche, R., Faugeras, O., Luong, Q.T.: A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. AI **78**(1-2) (1995) 87–119

[3] Fitzgibbon, A.W., Zisserman, A.: Automatic 3D model acquisition and generation of new images from video sequences. In: EUSIPCO. (1998) 1261–1269
[4] Pollefeys, M., Gool, L.V., Vergauwen, M., Cornelis, K., Verbiest, F., Tops, J.: Image-based 3D acquisition of archaeological heritage and applications. In: Proceedings of the 2001 conference on Virtual reality, archeology, and cultural heritage. (2001) 255–262
[5] Gibson, S., Hubbold, R.J., Cook, J., Howard, T.L.J.: Interactive reconstruction of virtual environments from video sequences. Computers and Graphics **27** (2003) 293–301
[6] Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision (DARPA). In: Proceedings of the 1981 DARPA Image Understanding Workshop. (1981) 121–130
[7] Shi, J., Tomasi, C.: Good features to track. In: CVPR. (1994) 593–600
[8] Lowe, D.G.: Object recognition from local scale-invariant features. In: ICCV. (1999) 1150
[9] Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV **60**(2) (2004) 91–110
[10] Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. TPAMI **02** (2005) 257
[11] Brown, M., Lowe, D.G.: Recognising panoramas. In: ICCV. (2003) 1218–1225
[12] Gordan, I., Lowe, D.G.: Scene modelling, recognition and tracking with invariant image features. In: ISMAR. (2004) 110–119
[13] Beis, J.S., Lowe, D.G.: Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In: CVPR. (1997) 1000
[14] Ke, Y., Sukthankar, R.: PCA-SIFT: A more distinctive representation for local image descriptors. In: CVPR. (2004) 506–513
[15] Pollefeys, M., Gool, L.V., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., Koch, R.: Visual modeling with a hand-held camera. IJCV **59**(3) (2004) 207–232
[16] Fitzgibbon, A.W., Cross, G., Zisserman, A.: Automatic 3D model construction for turn-table sequences. In Koch, R., Gool, L.V., eds.: Proceedings of SMILE Workshop on Structure from Multiple Images in Large Scale Environments. Volume 1506. (1998) 154–170
[17] Gibson, S., Cook, J., Howard, T.L.J., Hubbold, R.J., Oram, D.: Accurate camera calibration for off-line, video-based augmented reality. In: ISMAR. (2002) 37
[18] Repko, J., Pollefeys, M.: 3D models from extended uncalibrated video sequence: addressing key-frame selection and projective drift. In: Fifth International Conference on 3-D Digital Imaging and Modeling. (2005) 150–157
[19] Dey, T.K., Goswami, S.: Provable surface reconstruction from noisy samples. In: Proceedings of 20th ACM-SIAM Symposium on Computational Geometry. (2004) 330–339

Surface Fitting to Curves with Energy Control

Wen-Ke Wang[1,2], Hui Zhang[1], Jun-Hai Yong[1], and Jia-Guang Sun[1,2]

[1] School of Software, Tsinghua University, Beijing, P.R. China
[2] Department of Computer Science and Technology, Tsinghua University, Beijing, P.R. China

Abstract. An algorithm for surface fitting to curves with energy control is proposed in this paper. Given four boundary curves and a set of unorganized curves, we impose the constrained energy on the desired surface, and covert the minimum energy problem into a linear equation system of the control points of the surface. We prove that there is one unique solution of this equation system. The proposed algorithm is independent of the coordinate system, and experience shows that the resultant surface is fair.

1 Introduction

B-spline surfaces play an important role in Computer Aided Geometric Design (CAGD) [1,2]. There are various methods to construct a B-spline surface by fitting a set of B-spline curves [3,4,5]. Unfortunately, most of the existing methods require that the curves are in order. A novel algorithm for fitting a B-spline surface from a set of unorganized curves is proposed by Maekawa et al. [6]. Maekawa et al.'s algorithm needs the input of four boundary curves, and a set of unorganized curves that can take arbitrary orientation and possibly intersect each other. The main idea of the algorithm is as follows.

Firstly, a Coons surface is constructed by interpolating four boundary curves. Then the parametrization of each unorganized curve can be obtained by projecting itself onto the surface. Assuming that the curves are all on the final surface, they derive a linear equation system of the control points of the surface. The fitted surface can be obtained simply by solving the linear equation system.

Due to the local support property of B-spline, the matrix of the equation system is often singular when the distribution of the input curves is sparse or non-uniform. In this case, Maekawa et al. [6] use Singular Value Decomposition [7] (SVD) to get one of the solution of the equation system in the least square sense. However, their method may fail to get a satisfactory surface since it doesn't consider the geometry and physical character of the surface. Ferguson et al. [8] employ the null space of $\mathbf{A}$ (i.e., the space that satisfies $\mathbf{AX} = \mathbf{0}$) which is the matrix of the equation system in the singular case. Their algorithm selects a component from the null space to obtain a surface with a second derivative whose square integrated over the domain is minimized. Unfortunately, Ferguson et al. do not give a method to solve this optimization problem, and they do not prove that their algorithm can get one unique solution either. In fact, the square integrated over the domain is a kind of energy. The constrained energy is widely used in surface modeling to control the shape of the surface or to fair the surface [9,10]. In this paper, we impose a more general constrained energy on the surface to be fitted to improve the fairness of the surface. Compared with the method proposed in Ref. [6], the user can get more degrees of freedom to control the shape of the fitted surface.

The major contributions of our work are as follows.

- We provide a linear equation system of the control points of the surface to be fitted for minimizing the energy of the surface.
- We prove that there is one unique solution of the equation system.
- With the constrained energy, the fitted surface is fairer than the surface obtained by the SVD method as proposed in Ref. [6] in most cases, and it is independent of the coordinate system.

2 The Energy of a B-Spline Surface

A B-spline surface is defined as

$$\mathbf{S}(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(u)N_{j,q}(v)\mathbf{P}_{i,j}, \tag{1}$$

where $N_{i,p}(u)$ is the ith B-spline basis function of p-degree in u direction and $N_{j,q}(v)$ is the jth basis function of q-degree in v direction. $\mathbf{P}_{i,j}$ are the control points of the surface.

In this paper, we use the energy proposed in Ref. [9]. The energy formula is

$$e = \int_0^1\int_0^1 \left(\begin{array}{l} \alpha_{1,1}\left(\dfrac{\partial \mathbf{S}}{\partial u}\right)^2 + \alpha_{2,2}\left(\dfrac{\partial \mathbf{S}}{\partial v}\right)^2 \\ +\beta_{1,1}\left(\dfrac{\partial^2 \mathbf{S}}{\partial u^2}\right)^2 + \beta_{2,2}\left(\dfrac{\partial^2 \mathbf{S}}{\partial v^2}\right)^2 \\ +2\alpha_{1,2}\dfrac{\partial \mathbf{S}}{\partial u}\dfrac{\partial \mathbf{S}}{\partial v} + 2\beta_{1,2}\left(\dfrac{\partial^2 \mathbf{S}}{\partial u \partial v}\right)^2 \end{array} \right) dudv. \tag{2}$$

This kind of energy considers the geometric property and the elastically deformation of the surface. Generally speaking, by minimizing the above energy of the surface, we will build a shape which naturally attempts to resist stretching and bending so that the fairness of the fitted surface will be good [9]. The coefficients in Formula(2) represent for the material characteristic of the surface. We set $\alpha_{1,2} = 0$ and other coefficients are all larger than zero. In this setting, we can get the following theorem.

Theorem 1. *The energy of a surface is no less than zero, and is equal to zero if and only if every control point of the surface equals to each other.*

Proof. We know that the energy of a surface can't be negative from the setting of the coefficients of the energy. If the energy is equal to zero, we get $\frac{\partial \mathbf{S}}{\partial u} = 0$ and $\frac{\partial \mathbf{S}}{\partial v} = 0$ from Formula(2), which means that the surface $\mathbf{S}(u, v) = \mathbf{C}$, where $\mathbf{C}$ is a constant point. That is, the energy of a surface is equal to zero if and only if the surface degenerates into a point, which means that every control point of the surface equals to each other. □

3 The Optimization Problem and Its Solution

3.1 The Overview of the Algorithm

We summarize the algorithm for surface fitting proposed in Ref. [6] as follows. The input of the algorithm is four boundary curves and a set of unorganized curves; all these curves are in B-spline form. The output of the algorithm is a B-spline surface. The main steps are as follows.

(1) Construct a Coons surface by bilinearly blending the given four boundary curves.

(2) Subdivide the input B-spline curves into Bézier segments and project the two end points of each Bézier segment onto the Coons surface. The parameters of these two projected points are denoted as (u_1, v_1) and (u_2, v_2). The straight line between (u_1, v_1) and (u_2, v_2) is assumed to be the parameterization of this Bézier segment on the final surface.

(3) The final surface is in B-spline form shown in Eq. (1). Subdivide the final surface into Bézier surface segments. The input B-spline curves should be on the final surface. Suppose the ith Bézier curve on the jth Bézier surface segments, then an equation can be written as

$$\mathbf{d}_i\,(t) = \mathbf{r}_j\,(u(t), v(t))\,, \tag{3}$$

where $\mathbf{d}_i\,(t) = \sum_{k=0}^{m} \mathbf{d}_k B_{k,m}(t)$ is the ith Bézier curve and

$$\mathbf{r}_j(u, v) = \sum_{k_1=0}^{n_1} \sum_{k_2=0}^{n_2} \mathbf{r}_{k_1,k_2} B_{k_1,n_1}(u) B_{k_2,n_2}(v)$$

is the jth Bézier surface. A linear equation system is obtained from Eq. (3); the unknowns are the control points of the Bézier surface corresponding to the Bézier curve, which are linear combinations of the control points of the surface to be fitted. For each Bézier curve, a linear equation system is obtained, and finally a whole linear equation system is formed by combining all equation systems. We denote the whole linear equation system as

$$\mathbf{AX} = \mathbf{B}, \tag{4}$$

where $\mathbf{X}$ are the unknown control points of the surface to be fitted.

Since the four boundary curves are given, Eq. (4) becomes

$$\mathbf{A}_1\mathbf{X}_1 = \mathbf{B} - \mathbf{A}_2\mathbf{X}_2, \tag{5}$$

where $\mathbf{X}_2$ are the boundary control points and $\mathbf{X}_1$ are the other control points.

(4) Solve this equation system to get the surface.

As mentioned in Ref. [6], the equation system (5) is usually overdetermined so that no unique solution can be found. In this case, Maekawa et al. [6] suggest to use the SVD method to get a solution of the system (5) in the least square sense. In fact, due to the local support property of B-spline, we even can't get one unique solution of the system in the least square sense when the distribution of the input curves is sparse or non-uniform. On the other hand, when the system (5) is underdetermined, there may also be many solutions. Therefore, we need to choose one. In fact, we can use SVD to get one of the solution of the system (5) in any case. When the system (5) is overdetermined, we get a solution in the least square sense. While when the system (5) is underdetermined, we get an exact solution. Unfortunately, the SVD method doesn't consider the geometry and physical character of the surface and often can't get a satisfactory surface as shown in Section 4. In Section 3.2, we will impose the constrained energy on the surface to be fitted, which results in an optimization problem. Finally, we will get one unique solution for this optimization problem and the fitted surface is fairer than that obtained by SVD.

3.2 The Solution of the Equation System Based on the Constrained Energy

In this section, we first get the energy of a surface in matrix form for calculating the energy conveniently. We rewrite the control points of the surface in the form

$$\mathbf{X} = \left[\mathbf{p}_{0,0}, \mathbf{p}_{0,1}, \dots, \mathbf{p}_{0,n}, \mathbf{p}_{1,0}, \dots, \mathbf{p}_{m,n}\right]^T. \tag{6}$$

Since the boundary of the surface is fixed, Eq. (6) can be rewritten as

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix}, \tag{7}$$

where $\mathbf{X}_2$ are the boundary control points and $\mathbf{X}_1$ are the other control points. Correspondingly, the bases of the surface are given in the following formula:

$$\mathbf{N} = \begin{bmatrix} \mathbf{N}_1 \\ \mathbf{N}_2 \end{bmatrix} \tag{8}$$

The surface is rewritten as $\mathbf{S} = \mathbf{N}_1^T\mathbf{X}_1 + \mathbf{N}_2^T\mathbf{X}_2$. Then, we get the derivatives of the surface

$$\begin{aligned}
\frac{\partial \mathbf{S}}{\partial u} &= \frac{\partial \mathbf{N}_1^T}{\partial u}\mathbf{X}_1 + \frac{\partial \mathbf{N}_2^T}{\partial u}\mathbf{X}_2,\\
\frac{\partial \mathbf{S}}{\partial v} &= \frac{\partial \mathbf{N}_1^T}{\partial v}\mathbf{X}_1 + \frac{\partial \mathbf{N}_2^T}{\partial v}\mathbf{X}_2,\\
\frac{\partial^2 \mathbf{S}}{\partial u^2} &= \frac{\partial^2 \mathbf{N}_1^T}{\partial u^2}\mathbf{X}_1 + \frac{\partial^2 \mathbf{N}_2^T}{\partial u^2}\mathbf{X}_2,\\
\frac{\partial^2 \mathbf{S}}{\partial v^2} &= \frac{\partial^2 \mathbf{N}_1^T}{\partial v^2}\mathbf{X}_1 + \frac{\partial^2 \mathbf{N}_2^T}{\partial v^2}\mathbf{X}_2,\\
\frac{\partial^2 \mathbf{S}}{\partial u\partial v} &= \frac{\partial^2 \mathbf{N}_1^T}{\partial u\partial v}\mathbf{X}_1 + \frac{\partial^2 \mathbf{N}_2^T}{\partial u\partial v}\mathbf{X}_2.
\end{aligned} \tag{9}$$

Substituting Eqs. (9) into Formula(2), we get the energy expression in the form

$$\begin{aligned} e &= \mathbf{X}_1^T\mathbf{H}_1\mathbf{X}_1 + 2\mathbf{X}_2^T\mathbf{H}_2\mathbf{X}_1 + \mathbf{X}_2^T\mathbf{H}_3\mathbf{X}_2 \\ &= \mathbf{X}_1^T\mathbf{H}_1\mathbf{X}_1 + 2\mathbf{g}^T\mathbf{X}_1 + c \end{aligned}, \tag{10}$$

where $\mathbf{g}^T = \mathbf{X}_2^T\mathbf{H}_2$, $c = \mathbf{X}_2^T\mathbf{H}_3\mathbf{X}_2$, and

$$\mathbf{H}_1 = \int_0^1\!\!\int_0^1 \left(\begin{array}{l} \alpha_{1,1}\frac{\partial \mathbf{N}_1}{\partial u}\frac{\partial \mathbf{N}_1^T}{\partial u} + \alpha_{2,2}\frac{\partial \mathbf{N}_1}{\partial v}\frac{\partial \mathbf{N}_1^T}{\partial v} \\ +\beta_{1,1}\frac{\partial^2 \mathbf{N}_1}{\partial u^2}\frac{\partial^2 \mathbf{N}_1^T}{\partial u^2} + \beta_{2,2}\frac{\partial^2 \mathbf{N}_1}{\partial v^2}\frac{\partial^2 \mathbf{N}_1^T}{\partial v^2} \\ +2\alpha_{1,2}\frac{\partial \mathbf{N}_1}{\partial u}\frac{\partial \mathbf{N}_1^T}{\partial v} + 2\beta_{1,2}\frac{\partial^2 \mathbf{N}_1}{\partial u\partial v}\frac{\partial^2 \mathbf{N}_1^T}{\partial u\partial v} \end{array}\right) dudv,$$

$$\mathbf{H}_2 = \int_0^1\!\!\int_0^1 \left(\begin{array}{l} \alpha_{1,1}\frac{\partial \mathbf{N}_2}{\partial u}\frac{\partial \mathbf{N}_1^T}{\partial u} + \alpha_{2,2}\frac{\partial \mathbf{N}_2}{\partial v}\frac{\partial \mathbf{N}_1^T}{\partial v} \\ +\beta_{1,1}\frac{\partial^2 \mathbf{N}_2}{\partial u^2}\frac{\partial^2 \mathbf{N}_1^T}{\partial u^2} + \beta_{2,2}\frac{\partial^2 \mathbf{N}_2}{\partial v^2}\frac{\partial^2 \mathbf{N}_1^T}{\partial v^2} \\ +2\alpha_{1,2}\frac{\partial \mathbf{N}_2}{\partial u}\frac{\partial \mathbf{N}_1^T}{\partial v} + 2\beta_{1,2}\frac{\partial^2 \mathbf{N}_2}{\partial u\partial v}\frac{\partial^2 \mathbf{N}_1^T}{\partial u\partial v} \end{array}\right) dudv,$$

$$\mathbf{H}_3 = \int_0^1\!\!\int_0^1 \left(\begin{array}{l} \alpha_{1,1}\frac{\partial \mathbf{N}_2}{\partial u}\frac{\partial \mathbf{N}_2^T}{\partial u} + \alpha_{2,2}\frac{\partial \mathbf{N}_2}{\partial v}\frac{\partial \mathbf{N}_2^T}{\partial v} \\ +\beta_{1,1}\frac{\partial^2 \mathbf{N}_2}{\partial u^2}\frac{\partial^2 \mathbf{N}_2^T}{\partial u^2} + \beta_{2,2}\frac{\partial^2 \mathbf{N}_2}{\partial v^2}\frac{\partial^2 \mathbf{N}_2^T}{\partial v^2} \\ +2\alpha_{1,2}\frac{\partial \mathbf{N}_2}{\partial u}\frac{\partial \mathbf{N}_2^T}{\partial v} + 2\beta_{1,2}\frac{\partial^2 \mathbf{N}_2}{\partial u\partial v}\frac{\partial^2 \mathbf{N}_2^T}{\partial u\partial v} \end{array}\right) dudv.$$

The above three matrices (i.e. $\mathbf{H}_1$, $\mathbf{H}_2$ and $\mathbf{H}_3$) can be calculated by a numerical method, such as Gaussian quadrature [11].

It is obvious that the matrix $\mathbf{H}_1$ is symmetric and semi-positive. In fact, we can prove that the matrix $\mathbf{H}_1$ is positive definite.

Theorem 2. *The matrix $\boldsymbol{H}_1$ in Eq. (10) is positive definite.*

Proof. If $\exists \mathbf{X}_1 \neq \mathbf{0}$, $\mathbf{X}_1^T\mathbf{H}_1\mathbf{X}_1 = 0$. We set $\mathbf{X}_2 = \mathbf{0}$. We know that the energy of the surface is larger than zero from Theorem 1, that is $e > 0$. However, when we substitute $\mathbf{X}_1$ and $\mathbf{X}_2$ into Eq. (10), we get

$$\begin{aligned} e &= \mathbf{X}_1^T\mathbf{H}_1\mathbf{X}_1 + 2\mathbf{X}_2^T\mathbf{H}_2\mathbf{X}_1 + \mathbf{X}_2^T\mathbf{H}_3\mathbf{X}_2 \\ &= 0 + 2\mathbf{0}^T\mathbf{H}_2\mathbf{X}_1 + \mathbf{0}^T\mathbf{H}_3\mathbf{0} \\ &= 0. \end{aligned}$$

It conflicts with the former inequation ($e > 0$). Since $\mathbf{H}_1$ is semi-positive, we get $\forall \mathbf{X}_1 \neq \mathbf{0}$, $\mathbf{X}_1^T\mathbf{H}_1\mathbf{X}_1 > 0$. □

Now we consider the solution of the linear equation system (5). Suppose

$$\mathbf{X}_1 = \mathbf{Y}_1 + \mathbf{Z}_1, \tag{11}$$

where $\mathbf{Y}_1$ is obtained by SVD, and $\mathbf{Z}_1$ is comprised of the bases of the null space of $\mathbf{A}_1$. Substituting Eq. (11) into Eq. (10), we get the expression of the energy shown as

$$\begin{aligned} e &= (\mathbf{Y}_1 + \mathbf{Z}_1)^T\,\mathbf{H}_1\,(\mathbf{Y}_1 + \mathbf{Z}_1) + 2\mathbf{g}^T\,(\mathbf{Y}_1 + \mathbf{Z}_1) + c \\ &= \mathbf{Z}_1^T\mathbf{H}_1\mathbf{Z}_1 + 2\mathbf{f}^T\mathbf{Z}_1 + c_1 \end{aligned}, \tag{12}$$

where $\mathbf{f}^T = \mathbf{Y}_1^T\mathbf{H}_1 + \mathbf{g}^T$ and $c_1 = \mathbf{Y}_1^T\mathbf{H}_1\mathbf{Y}_1 + 2\mathbf{g}^T\mathbf{Y}_1 + c$.

Now we need to minimize the following functions:

$$\begin{aligned} &e_1 = \frac{1}{2}\mathbf{Z}_1^T\mathbf{H}_1\mathbf{Z}_1 + \mathbf{f}^T\mathbf{Z}_1 \\ &\mathbf{A}_1\mathbf{Z}_1 = \mathbf{0} \end{aligned} \tag{13}$$

System(13) is a linearly constrained quadratic minimization problem, and it can be solved by many existing methods. Most of the methods require that the rows of $\mathbf{A}_1$ are linearly independent [10,12] which is not satisfied in many cases in surface fitting problem, unfortunately. In this paper, we solve this optimization problem by using the null space of $\mathbf{A}_1$ to avoid the problem that the rows of $\mathbf{A}_1$ are linearly dependent. Considering that $\mathbf{Z}_1$ is comprised of the bases of the null space of $\mathbf{A}_1$, we decompose $\mathbf{Z}_1$ into

$$\mathbf{Z}_1 = \mathbf{CW}, \tag{14}$$

where $\mathbf{C}$ is a set of orthogonal bases of the null space of $\mathbf{A}_1$, and the bases can be obtained as the by-product of SVD. $\mathbf{W}$ is the coefficient vector.

Substituting Eq. (14) into Eq. (13), we get another expression of e_1 shown as

$$\begin{aligned} e_1 &= \frac{1}{2}\mathbf{W}^T\mathbf{C}^T\mathbf{H}_1\mathbf{CW} + \mathbf{f}^T\mathbf{CW} \\ &= \frac{1}{2}\mathbf{W}^T\mathbf{QW} + \mathbf{h}^T\mathbf{W} \end{aligned}, \tag{15}$$

where $\mathbf{Q} = \mathbf{C}^T\mathbf{H}_1\mathbf{C}$ and $\mathbf{h}^T = \mathbf{f}^T\mathbf{C}$. Eq. (15) is a quadratic minimization problem with no constrained.

Since $\mathbf{H}_1$ is symmetric, $\mathbf{Q}$ is symmetric. In the following, we prove that the matrix $\mathbf{Q}$ is positive definite. In order to prove this theorem, we first prove the following theorem.

Theorem 3. *$\boldsymbol{Y} = \boldsymbol{CX} = \boldsymbol{0}$, if and only if $\boldsymbol{X} = \boldsymbol{0}$, where $\boldsymbol{C}$ is a set of orthogonal bases of the null space of $\boldsymbol{A}_1$.*

Proof. It is obvious that $\mathbf{Y} = \mathbf{0}$ when $\mathbf{X} = \mathbf{0}$. Now we assume that $\mathbf{X} = [a_1, ..., a_r]^T \neq \mathbf{0}$. $\mathbf{C} = \begin{bmatrix} c_{1,1}, \ldots, c_{1,r} \\ \vdots, \ddots, \vdots \\ c_{n,1}, \ldots, c_{n,r} \end{bmatrix} = [\mathbf{c}_1, \ldots, \mathbf{c}_r]$, then we get $\mathbf{Y} = a_1\mathbf{c}_1 + \ldots + a_r\mathbf{c}_r$. Since $\mathbf{c}_1, \ldots, \mathbf{c}_r$ is a set of linearly independent orthogonal bases of the null space of $\mathbf{A}_1$, $\mathbf{Y} \neq \mathbf{0}$. □

Theorem 4. *The matrix $\boldsymbol{Q}$ in Eq. (15) is positive definite.*

Proof. It is obvious that $\mathbf{X}^T\mathbf{Q}\mathbf{X} = 0$ when $\mathbf{X} = \mathbf{0}$. When $\mathbf{X} \neq \mathbf{0}$, we know that $\mathbf{Y} = \mathbf{CX} \neq \mathbf{0}$ from Theorem 3. Then we get $\mathbf{X}^T\mathbf{Q}\mathbf{X} = \mathbf{X}^T\mathbf{C}^T\mathbf{H}_1\mathbf{C}\mathbf{X} = \mathbf{Y}^T\mathbf{H}_1\mathbf{Y} > 0$ from Theorem 2. □

4 Examples

In this section, we give some examples to illustrate the effect of the constrained energy imposed on the surface. A set of curves, including four boundary curves and three other curves, is shown in Fig.1(a). The three orthogonal arrows indicate the object-oriented coordinate system. Fig.1(b) and Fig.1(c) show two fitted surfaces. Fig.1(b) shows the surface obtained by the SVD method. Fig.1(c) shows the surface obtained by our optimization. In our experiment, the coefficients of the energy are as follows. $\alpha_{1,1}=1, \alpha_{1,2}=0, \alpha_{2,2}=1, \beta_{1,1}=1, \beta_{1,2}=10, \beta_{2,2}=1$.

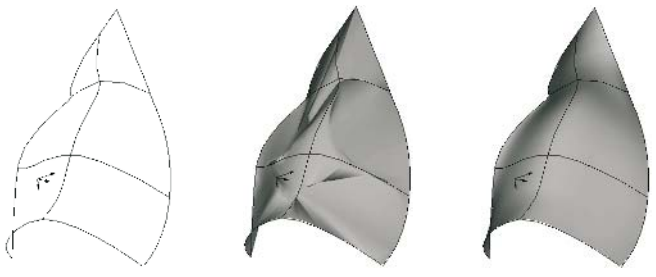

(a) (b) (c)

Fig. 1. Input curves and the fitted surfaces. (a) Input curves. (b) Surface obtained by the SVD method. (c) Surface obtained by the constrained energy method.

Since the basis functions of the B-spline surface are fixed, the degree of the smoothness of the two fitted surfaces (shown in Fig.1(b) and Fig.1(c)) are the same. But it is obvious that the surface in Fig.1(c) is much fairer than that shown in Fig.1(b). It is easy to give a qualitative analysis that why the shape of the surface shown in Fig.1(b) is not good. The SVD solution doesn't consider the geometric property and the elastically deformation of the surface. It only gets the solution with the minimal norm. Based on this reason, the surface obtained by the SVD method tends to the original point so that the norm of the control points will be small. While the energy of the surface considers the geometric property and the elastically deformation of the surface and it tries to get a surface which resist stretching and bending. As a result, a fairer surface will be obtained.

On the other hand, since the SVD method only gets the solution with minimal norm, the fitted surface is dependent on the coordinates of the input curves. On the contrary, the energy of the surface is independent of the coordinate system, which means that the shape of the fitted surface is unchanged while the curves undergoing translation and rotation. To illustrate this instance, we translate each curve by a distance as shown in Fig.2(a)(The translation can be seen easily by comparing the coordinate system shown in Fig.1(a) and Fig.2(a)). The two fitted surfaces are shown in Fig.2(b) and Fig.2(c).

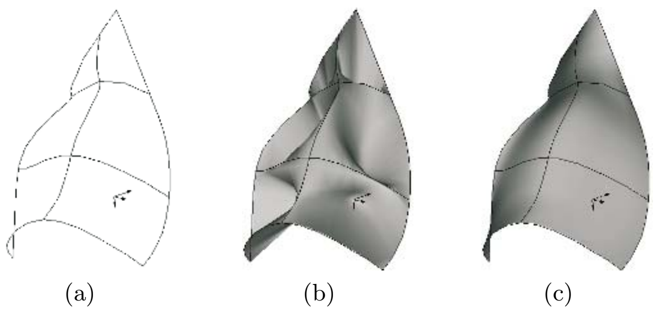


Fig. 2. Curves translated by a distance and the fitted surfaces. (a) Curves translated by a distance. (b) Surface obtained by the SVD method. (c) Surface obtained by the constrained energy method.

The fitted surface obtained by the SVD method, as shown in Fig.2(b), is much different from that shown in Fig.1(b). While the surface obtained by the constrained energy method, as shown in Fig.2(c), looks the same as that shown in Fig.1(c). In fact, the surface in Fig.2(c) is just a translation of the surface in Fig.1(c). In most cases, the designer considers the shape of the curves much more than the coordinates of the curves, so the surface obtained by the constrained energy method is more suitable for the designer.

Next, we will give a more complicated example. A set of curves, including four boundary curves and four other curves, is shown in Fig.3(a). Fig.3(b) and Fig.3(c) show two fitted surfaces. Fig.3(b) shows the surface obtained by the SVD method and Fig.3(c) shows the surface obtained by our optimization. The surface shown in Fig.3(c) is fairer than that shown in Fig.3(b).

At the end of this section, we show the difference between the energy used in Ref. [8] and the energy used in this paper. If we use the energy proposed in Ref.

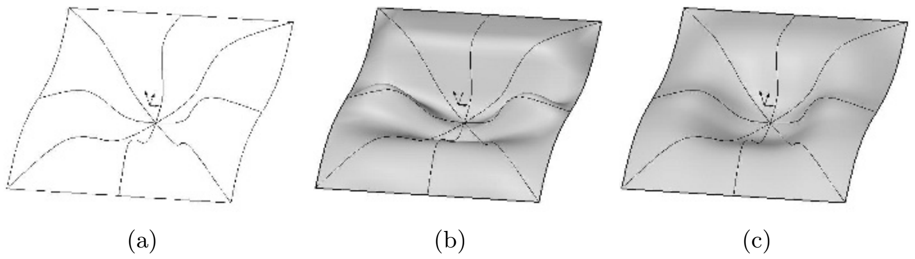


Fig. 3. Another example. (a) Input curves. (b) Surface obtained by the SVD method. (c) Surface obtained by the constrained energy method.

[8], we set $\alpha_{1,1} = 0, \alpha_{1,2} = 0, \alpha_{2,2} = 0, \beta_{1,1} = 1, \beta_{1,2} = 0, \beta_{2,2} = 1$ in Formula(2).

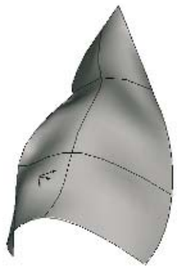


Fig. 4. Surface obtained by the constrained energy proposed in [5]

The fitted surface is shown in Fig.4. It is not as fair as that in Fig.1(c). The energy used in this paper (proposed in Ref. [9]) can provide the user more degrees of freedom to control the shape of the surface to be fitted. Generally speaking, $\alpha_{1,1}$ may affect the length of the iso-parameter curves of the surface in u direction, and $\alpha_{2,2}$ may affect the length in v direction. $\alpha_{1,2}$ may affect the area of the surface. $\beta_{1,1}$, $\beta_{1,2}$ and $\beta_{2,2}$ may affect the curvature of the surface. Unfortunately, we don't know the exact relationship between these parameters and the fitted surface, which will be our future work.

5 Conclusions

In this paper, we impose an energy item on the surface that fits a set of unorganized curves. We also give an equation system of the control points of the surface in order to minimize the energy of the surface and prove that there is one unique solution of this equation system. With the constrained energy introduced, the fitted surface is fairer than that obtained by SVD in most cases and it is also unchanged while the curves undergoing translation and rotation.

We point out that the calculation of the energy matrix is time consuming, but it is only computed just once in the whole algorithm. In some cases that the surface can't interpolate the curves, we can replace the Coons surface by the fitted surface and repeat the algorithm to reduce the fitting error. In this iterative process, the energy matrix does not need to be recomputed.

Acknowledgements

We would like to thank Professor Maekawa for many helpful discussions. The research was supported by Chinese 973 Program (2004CB719400), and the National Science Foundation of China (60403047). The third author was supported by the project sponsored by a Program for New Century Excellent Talents in University (NCET-04-0088).

References

1. Piegl, L.A., Tiller, W.: The NURBS Book. 2nd ed. Springer. New York. 1997
2. Farin, G.: Curves and surfaces for CAGD. 5th ed. San Francisco: Morgan Kaufmann. 2002
3. Woodward, C.: Skinning techniques for interactive B-spline surface interpolation. Computer-Aided Design. 11 (1988) 441-451
4. Coons, S.A.: Surfaces for computer-aided design of space forms. Massachusetts Institute of Technology, Cambridge, MA. 1967
5. Gordon, W.: Spline-blended surface interpolation through curve networks. Jour. Math. Mech. 8 (1969) 931-952
6. Maekawa, T., Ko, K.H.: Surface construction by fitting unorganized curves. Graphical Models. 64 (2002) 316-332
7. Golub, G.H., Reinsch, C.: Singular value decomposition and least squares solution. Numer. Math. 14 (1970) 403-420
8. Ferguson, D.R., Grandine, T.A.: On the Construction of Surfaces Interpolating Curves: I. A Method for Handling Nonconstant Parameter Curves. ACM Transactions on Graphics. 9 (1990) 212-225
9. Celniker, G., Gossard, D.: Deformable curve and surface finite elements for free-form shape design. Computer Graphics (ACM). 25 (1991) 257-266
10. Welch, W., Witkin, A.: Variational Surface Modeling. Computer Graphics (ACM). 26 (1992) 157-166
11. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes in C : The Art of Scientific Computing. 2nd ed. New York : Cambridge University Press. 1995
12. Yuan, Y.X.: Numerical methods for nonlinear programming. Shanghai Scientific & Technical Publishers Press. 1993 (in Chinese)

Blob Tracking with Adaptive Feature Selection and Accurate Scale Determination

Jingping Jia[1,2], David Feng[1,3], Yanmei Chai[2], Rongchun Zhao[1,2], and Zheru Chi[1,3]

[1] Department of Electronic and Information Engineering
The Hong Kong Polytechnic University, Hong Kong
[2] School of Computer Science and Engineering
Northwestern Polytechnical University, Xi'an 710072, P.R. China
[3] School of Information Technologies
University of Sydney, Australia

Abstract. We propose a novel color based tracking framework in which an object configuration and color feature are simultaneously determined via scale space filtration. The tracker can automatically select discriminative color feature that well distinguishes foreground from background. According to that feature, a likelihood image of the target is generated for each incoming frame. The target's area turns into a blob in the likelihood image. The scale of this blob can be determined based on the local maximum of differential scale-space filters. We employ the QP_TR trust region algorithm to search for the local maximum of multi-scale normalized Laplacian filter of the likelihood image to locate the target as well as determine its scale. Based on the tracking results of sequence examples, the proposed method has been proven to be resilient to the color and lighting changes, be capable of describing the target more accurately and achieve much better tracking precision.

1 Introduction

Object tracking in image sequences is a key issue in many computer vision applications such as video surveillance, perceptual user interfaces, object-based video compression, etc. Two major components can be distinguished in a typical visual tracker: target representation and target localization. Gray scale values, shape information, colors, textures, velocity and acceleration are the commonly used object's features. Target localization is the procedure to locate the area that matches the object's feature best, and it can be addressed by particular optimization methods.

It is well known that the success or failure of object tracking is primarily dependent on how distinguishable the feature of an object is from its surroundings and background. In literatures, Shi and Tomasi [1] have pointed out that good features are as important as good tracking algorithms. The degree and discriminability to which a tracker can discriminate the object and background is directly related to the image features used. It is the ability to distinguish between object and background that is the most important. In [2] a fixed set of candidate features are assessed in terms of the variance ratio and the best N ones are chosen to produce the likelihood images for tracking. Bohyung Han[3] used PCA to extract the most discriminative feature from the feature set composed of every subset of the RGB and rgb color channels. Stern and Efros [4] chose the best features from 5 features spaces and switch amongst them to improve the tracking performance. All these methods focused on the determination of the best feature from a predefined feature set with finite number of features.

The tracking precision also has much to do with the description of the target's scaling. Most previous related work [5,6,7] addressed the scaling by working on some presumably possible discrete values of scale. However, discrete values cannot fit the complex movements of the target. Tyng-Luh and Hwann-Tzong[8] proposed to use a covariance ellipse to characterize an object and adopt a bivariate normal within the ellipse as the weighting function for the features. They dealt with the issue in a continuous manner but the features they used are just predefined color probability and edge density information, which are not guaranteed to be the best one. Collins [9] combined mean shift procedure with Lindeberg's theory to solve the scale problem, but he ignored the feature updating.

In this paper we extend the discrete feature set to a continuous feature space. The best feature is chosen out of this space in terms of the target-background class variance ratio. We also adapt Lindeberg's theory [10] of feature scale selection based on the local maxima of differential scale-space filters to deal with the description of the target and describe scaling in the continuous space.

Compared with the popular mean shift [6] method, trust region methods are more effective and can yield a better performance [8]. Based on the traditional trust region method, QP_TR algorithm [11] improves the way to get the object function's Hessian matrix and gradient, and achieves even better performance. In this paper, we combine the QP_TR method with the scale space theory and propose a new tracking algorithm in which tracking is implemented to search for local maxima of the multi-scale normalized Laplacian filter function with the QP_TR method. Details about QP_TR can be found in the Appendix.

2 Proposed Target Model

Our goal in this section is to develop an efficient method that automatically chooses the best feature for tracking. Features used for tracking only need be locally discriminative, in that the object only needs to be clearly separable from its immediate surroundings. We represent target appearance using histograms of color filter bank responses applied to R, G, and B pixel values within local image windows. This representation is chosen since it is relatively insensitive to variation in target appearance due to viewpoint, occlusion and non-rigidity. In [2] a fixed set of candidate features are evaluated and the best one is chosen to produce the likelihood image. Based upon [2] but different from it, we extend the candidate features in a continuous manner.

In this paper, the candidate features are composed of linear combinations of R, G, B pixel values. Specifically, we have chosen the following candidate feature:

$$F = \omega_1 \mathrm{R} + \omega_2 \mathrm{G} + \omega_3 \mathrm{B} \quad \omega_i \in \mathbf{R} \tag{1}$$

where F is the linear combination of R, G, and B with real coefficients except for $(\omega_1, \omega_2, \omega_3) = (0,0,0)$. Many common features from the literatures are included in the candidate space, such as the raw values of R, G, and B, intensity R+G+B, approximate chrominance features such as R-B, and so-called excess color features such as 2G-R-B. All features are normalized into the range 0 to 255 and discretized into histograms of length 255.

We follow [2] to use a "center-surround" approach to sampling pixels covering the object and background. A rectangular set of pixels covering the object is chosen to represent the object pixels, while a larger surrounding ring of pixels is chosen to represent the background. For an inner rectangle of dimensions $h_t \times w_t$ pixels, an outer margin of width $\mathbf{max}(h_t, w_t)$ pixels forms the background sample. We use the object pixels to get the target histogram H_{obj} for a candidate feature and the background pixels to get the background histogram H_{bg}. We form an empirical discrete probability distribution $p(i), i = 1...255$ for the object, and $q(i)$ for the background by normalizing each histogram by the number of elements in it.

From any candidate features, we can create a new feature that is "tuned" to discriminate between object and background pixels. The tuned feature is formed as the log likelihood ratio of the class conditional candidate feature distributions. The log likelihood ratio [3] of a feature value i is given by

$$L(i) = \mathbf{max}\left(-1, \mathbf{min}\left(1, \log \frac{\mathbf{max}(p(i), \delta)}{\mathbf{max}(q(i), \delta)}\right)\right) \tag{2}$$

where δ is a small value that prevents dividing by zero or taking the log of zero (we choose it to 0.001). The nonlinear log likelihood ratio maps object/background distributions into positive values for colors distinctive to the object and negative for colors associated with the background. Colors shared by both object and background tend towards zero. Back-projecting these log likelihood ratio values into the image produces a likelihood image suitable for tracking, as shown in Fig 1.

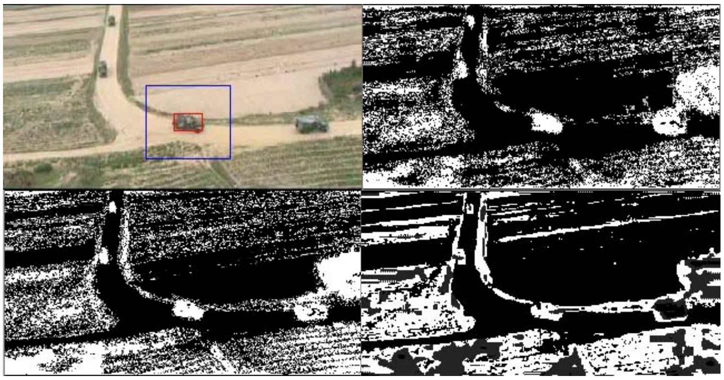


Fig. 1. Likelihood images by three methods, one frame from the "car" sequence

The separability that $L(i)$ induces between object and background classes can be measured using the two-class variance ratio. The variance of $L(i)$ with respect to object class distribution $p(i)$ is calculated as:

$$\operatorname{var}(L;p) = E\left[L^2(i)\right] - \left(E[L(i)]\right)^2 = \sum_i p(i)L^2(i) - \left[\sum_i p(i)L(i)\right]^2 \tag{3}$$

and similarly for background class distribution $q(i)$ [2]. Then the variance ratio can be defined as:

$$VR(L;p,q) \equiv \frac{\operatorname{var}(L;(p+q)/2)}{\operatorname{var}(L;p) + \operatorname{var}(L;q)} \tag{4}$$

The higher the ratio is, the wider the object and background are separated [2]. This means the triple of $(\omega_1, \omega_2, \omega_3)$ which produces the highest $VR(L;p,q)$, corresponds to the best feature. To get this best triple we can define an object function $\psi(\omega_1, \omega_2, \omega_3)$ that takes a triple $(\omega_1, \omega_2, \omega_3)$ as the parameter, calculates $p(i)$, $q(i)$, $L(i)$, and returns $-VR(L;p,q)$ .We call $\psi(\omega_1, \omega_2, \omega_3)$ the feature discriminability function. Apply the QP_TR method on $\psi(\omega_1, \omega_2, \omega_3)$ and we can get $(\omega_1, \omega_2, \omega_3)_{best}$. Introduce $(\omega_1, \omega_2, \omega_3)_{best}$ into (1) and we can get the best feature. Using the best feature we follow (2) to get the best "tuned" feature $L_{best}(i)$ which can be back-projected to produce the likelihood image.

Figure 1 shows the weight images calculated with three methods. The upper right one is by our method, the lower left one is by the method in [2], while the lower right is by that in [12].

Different from [2], we use real ω_i instead of integer ones. And we choose the best ω_i in the continuous space rather than from a finite set. This improvement enables us to get more approximate feature. For example, in Fig.1 the likelihood image using our method has a $VR = 1.89425$, which is larger than that of the likelihood image produced by method in [2], which is 1.60777. The corresponding triple $(\omega_1, \omega_2, \omega_3)$ is $(-5.02673, 4.9104, 0.959966)$.

3 Scale-Space Blob

During tracking, the size of the target will change with the distance to the camera. Tracking algorithm should adapt to this kind of change and describe it precisely. Most previous related work [5,6,7] addresses scaling by working on some presumably possible discrete values of scale. For example, for each frame the tracker in [6] tries two sizes of plus and minus ten percent of the previous size to guess the current size. However, it's difficult to adapt to the changes in size by using only these discrete values (We will show this later). Aiming at this shortcoming, we propose to solve the problem in the continuous scale space and describe the scaling in a continuous manner.

The work of Lindeberg [10] provided an elegant theory for selecting the best scale for describing structural features in an image. Given any continuous function $f : R^D \to R$ and a Gaussian kernel with scale t, $g : R^D \times R_+ \to R$, $g(\mathbf{x};t) = \frac{1}{(2\pi t)^{N/2}} e^{-(x_1^2+...+x_D^2)/(2t)}$, the scale-space representation of f is its convolution with g, i.e., $L : R^D \times R_+ \to R$, $L(\cdot;t) = g(\cdot;t) * f(\cdot)$ with various scale t. The γ-normalized derivative operator is defined as $\partial_\xi = t^{\gamma/2}\partial_x$. A very good property of the γ-normalized derivative of L is the perfect scale invariance as follows:

Consider two functions f and $\tilde{f}$ related by $f(\mathbf{x}) = \tilde{f}(\mathbf{x})$ and define the scale-space representation of f and $\tilde{f}$ in the two domains by

$$\begin{aligned} L(\cdot;t) &= g(\cdot;t) * f(\cdot) \\ \tilde{L}(\cdot;\tilde{t}) &= g(\cdot;\tilde{t}) * \tilde{f}(\cdot) \end{aligned} \tag{5}$$

where the spatial variables and the scale parameters are transformed according to

$$\begin{aligned} \tilde{\mathrm{x}} &= s\mathrm{x} \\ \tilde{t} &= s^2 t \end{aligned} \tag{6}$$

Then L and $\tilde{L}$ are related by $L(\mathbf{x};t) = \tilde{L}(\cdot;\tilde{t})$ and the m-th order spatial derivatives satisfy $\partial_{x^m} L(\mathbf{x};t) = s^m \partial_{\tilde{x}^m} \tilde{L}(\tilde{\mathbf{x}};\tilde{t})$. For γ-normalized derivatives defined in the two domains by

$$\begin{aligned} \partial_\xi &= t^{\gamma/2}\partial_x \\ \partial_{\tilde{\xi}} &= \tilde{t}^{\gamma/2}\partial_{\tilde{x}} \end{aligned} \tag{7}$$

we have $\partial_{\xi^m} L(\mathrm{x};t) = s^{m(1-\gamma)} \partial_{\tilde{\xi}^m} \tilde{L}(\tilde{\mathrm{x}};\tilde{t})$. From this relation it can be seen that, when $\gamma = 1$, $\partial_{\xi^m} L(\mathrm{x};t) = \partial_{\tilde{\xi}^m} \tilde{L}(\tilde{\mathrm{x}};\tilde{t})$. That is, if the γ-normalized derivative of f assumes a local maximum at $(\mathrm{x}_0;t_0)$ in the scale-space, the γ-normalized derivatives of $\tilde{f}$ will also assume a local maximum at $(s\mathrm{x}_0; s^2 t_0)$. Based on this property we can choose appropriate combination of the γ-normalized derivatives to determine the scale of some structure in the data.

A gray image can be seen as a two dimensional function. That is, $D = 2$, and $f : R^2 \to R$. When $\gamma = 1$, $\left(t^\gamma \nabla^2 L\right)^2 = \left(t\nabla^2 L\right)^2 = \left(t(L_{xx} + L_{yy})\right)^2$ reflects the details of blobs in an image. We call $\varphi(x,y,t) = \left(t(L_{xx}(x,y) + L_{yy}(x,y))\right)^2$ the multi-scale normalized Laplacian filter function. With different scale values t, $\varphi(x,y,t)$ achieves local maxima at blobs with different sizes. The gray image in Fig 2 contains two blobs. With $t = 90$, $\varphi(x, y, t)$ assumes the local maximum at the center of the smaller one, while with $t = 391$, $\varphi(x, y, t)$ assumes another local maximum at the center of the larger one. So the locations of the blobs as well as their scales can be determined by examining the local maxima of $\varphi(x, y, t)$ at various positions and scales.

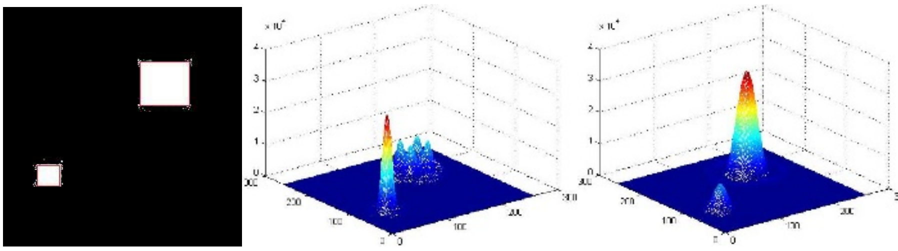


Fig. 2. For blobs with different sizes, $\varphi(x, y, t)$ assumes local maxima at different positions with different scales, Left: Original image Middle: The response of $\varphi(x, y, t)$ with $t = 90$ Right: The response of $\varphi(x, y, t)$ with $t = 391$

4 A Brief Outline of the Proposed Algorithm

As a brief summary, we track targets by searching its best feature, location and scale for the target in the current frame. For each incoming frame the likelihood image of target is calculated, and QP_TR trust region method is employed to search for the local maximum of $\phi(x, y, t)\psi(\omega_1, \omega_2, \omega_3)$ in the scale space of the likelihood image. The algorithm proceeds as follows:

1. Target initialization: get the width and height of the target, denoted as w_t and h_t. The width and height of the frame are w and h respectively. Using the method in section 3 to determine the best feature to use, record the corresponding triple $(\omega_1, \omega_2, \omega_3)_{best}$ and $L_{best}(i)$.
2. Resize the frame so that the ratio of width to height will be $\rho = wh_t / h / w_t$. If we hold w fixed the height of the frame will be w / ρ.
3. Initialize a vector $\mathbf{x}_0 = (\omega_{1prev}, \omega_{2prev}, \omega_{3prev}, x_{prev}, y_{prev}, t_{prev})^T$, where (x_{prev}, y_{prev}) is the center of the target in the previous frame, t_{prev} the scale parameter in the previous frame and $\left(\omega_{1prev}, \omega_{2prev}, \omega_{3prev}\right)$ the previous best feature. Set the initial trust region radius $\Delta_0 = 9$, the minimum radius $\Delta_{end} = 0.1$ and the maximum iteration $MAX_{iter} = 50$.
4. Run the QP_TR algorithm for $f(\mathbf{x}) = -\phi(x, y, t)\psi(\omega_1, \omega_2, \omega_3)$ and get $\mathbf{x}_{opt} = (\omega_{1opt}, \omega_{2opt}, \omega_{3opt}, x_{opt}, y_{opt}, t_{opt})^T$ which minimizes $f(\mathrm{x})$, where (x_{opt}, y_{opt}) is the center of the target in the current frame with t_{opt} its scale parameter and $\left(\omega_{1opt}, \omega_{2opt}, \omega_{3opt}\right)$ the optimal feature. Go to step 2) until the track is over.

5 Experiments, Results and Discussion

To verify the efficiency of our method, we apply it to many sequences and compare with other tracking algorithms. First, we evaluated the performance of feature selection of our novel method with [2], as shown in Fig 3. The number of evaluated features by our method is reduced a lot whereas the number is a fixed one in ref [2]. At the same time, the discriminability between the object and background has increased compared with ref [2]. From the results, we find that our method is more effective and efficient than that in [2] in term of the evaluated features and discriminability of the selected feature for object description, e.g., VR.

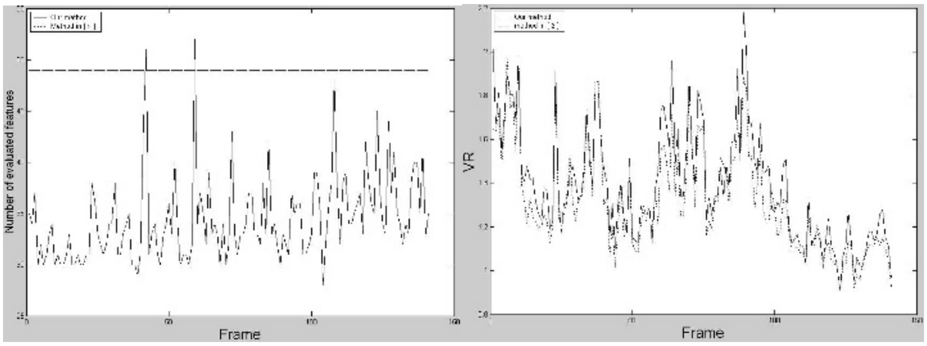


Fig. 3. The curves of the number of evaluated features and VR of the "car" sequence (from 1 to 142 frames)

To demonstrate our method's ability to adapt to the appearance changes, we test it on the "ellipse" sequence. In this sequence the ellipse is moving around the center of the image while its size and color change. An algorithm without feature updating, such as [6], will soon be distracted and shrinks. See fig 4. Our method searches for the best feature while locating the target, therefore, avoids this problem.

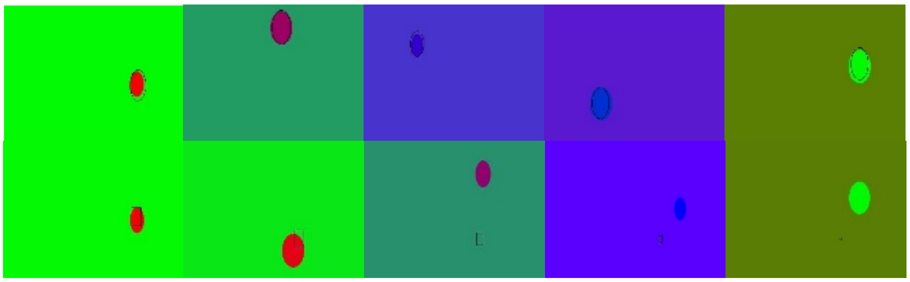


Fig. 4. Tracking results of "ellipse" sequence. Upper row: Results based on our method, Lower row: Results based on the method in [6].

Besides the above-mentioned good performance of feature extraction and updating for objects, the proposed method is proven to be accurate in describing the object and can yield better tracking result than those of refs [3, 5, 7, 13] by the following experimental results. Fig 5 shows the tracking result of the "pedestrian" sequence and performance comparison with the bandwidth mean shift tracker [7] In the bandwidth mean shift tracker, the scale parameter can take only three discrete values so that there is much error in describing the target's size, which results in error of the target localization (as shown in the lower row). The upper row shows our method's result. It can be seen that the black ellipse shrinks with the target and describes the target's size accurately.

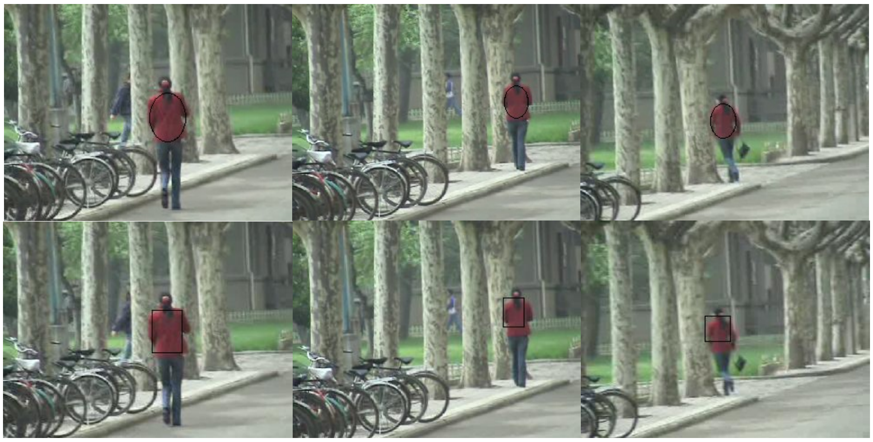


Fig. 5. Tracking results of a pedestrian. Upper row: Results based on our method. Lower row: Results of the bandwidth mean shift tracker. Only the 3rd, 254th and 341st frames are shown.

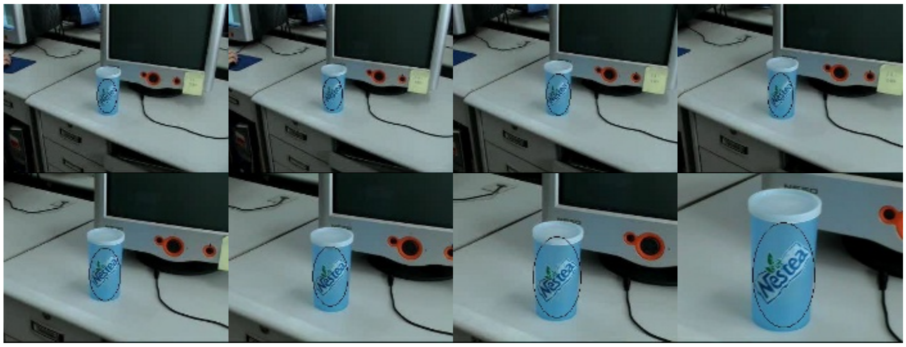


Fig. 6. Tracking results of the "cup" sequence. Only the 4th, 44th, 88th and 124th 168th, 204th, 244th and 332nd frame are shown.

Fig 6 is another example of our method's ability to cope with the zooming of target. The cup zooms in dramatically through the "cup" sequence. The proposed method still produces satisfactory description of the target.

6 Conclusion

In this paper, we proposed a new target tracking algorithm to address the problem of feature selection and target's scale determination, by the combination of Lindeberg's scale space theory with the QP_TR trust region method. Firstly, the best feature that best discriminates the target and background is automatically determined out of the continuous candidate feature space. Each target corresponds to a specific blob in the likelihood image. Then we introduce the multi-scale normalized Laplacian filter function to detect the blobs in gray images. Conceptually, the scale space is generated by convolving the likelihood image with a filter bank of spatial LOG filters. Explicit generation of these convolutions would be very expensive and only able to evaluate at finite discrete parameters. However, by using the QP_TR trust region method, we can search for the local maximum of the object function in the continuous scale space much more efficiently, where the local maximum corresponds to a blob, i.e. the target. The precise tracking of objects is fulfilled as searching the maxima of both the Laplacian filter function and the feature discriminability function. Experimental results have demonstrated our new algorithm's ability to adapt to objects' scale and appearance changes with much a better tracking precision.

Acknowledgment

This research is partially supported by GC, PolyU and ARC grants.

References

1. J. Shi and C. Tomasi. "Good features to track", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 1994, pp.593-600.
2. Robert T. Collins, Yanxi Liu, Marius Leordeanu, "Online selection of discriminative Tracking features", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.27, No.10, October 2005, pp. 1631-1643.
3. Bohyung Han, Larry Davis, "Object tracking by adaptive feature extraction", Proceedings of the 2004 International Conference on Image Processing, Singapore, 2004, Vol.3, pp.1504-1504.
4. H. Stern and B. Efros, "Adaptive color space switching for face tracking in multi-colored lighting environment", Proceedings of the International Conference on Automatic Face and Gester Recognition, Washington DC, USA, 2002, pp.249-254.
5. Comaniciu, D., Ramesh, V. and Meer, P., "Real-time tracking of non-rigid objects using mean shift", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Vol II, 2000, pp.142-149.
6. Comaniciu, D., Ramesh, V. and Meer, P., "Kernel-based object tracking", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.25, No.5, May 2003, pp.564-577.
7. Jingping Jia, Rongchun Zhao, "Tracking of objects in image sequences using bandwidth matrix mean shift algorithm", Proceedings of 7th International Conference on Signal Processing, vol 2, August 2004, pp. 918-921

8. Tyng-Luh Liu, Hwang-Tzong Chen, "Real-time tracking using trust-region methods", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.26, No.3, March 2004, pp. 397-402.
9. Robert T. Collins, "Mean-shift Blob Tracking through Scale Space", Proceedings of CVPR 2003, Madison, Wisconsin, vol 2, pp. 234-240, June 2003
10. Lindeberg, T., "Feature detection with automatic scale selection", International Journal of Computer Vision, Vol.30, No.2, Nov 1998, pp.79-116.
11. Frank Vanden Berghen, "Intermediate report on the development of an optimization code for smooth, continuous objective functions when derivatives are not available", http://www.optimization-online.org/DB_HTML/2003/08/704.html.
12. Bradski, G.R., "Computer vision face tracking for use in a perceptual user interface", IEEE Workshop on Applications of Computer Vision, Princeton, NJ, 1998, pp.214-219.
13. Jingping Jia, Yanning Zhang, etc, "Tracking of objects in image sequences using multi-freeness mean shift algorithm", Proceedings of the 4th International Conference on Machine Learning and Cybernetics, Vol.8, August 2005, pp. 5133-5138.

Appendix

QP_TR Trust Region Algorithm

The QP_TR trust region method can be used to solve the unconstrained minimization problem $\min_{\mathbf{x}\in\mathbf{V}} f(\mathbf{x})$, where V is a vector space and f is the objective function to be minimized. It derives its iterations by solving the corresponding optimization problem in a bounded region iteratively, which is called the trust region sub-problem. There are three parameters to specify: the initial trust region radius Δ_0, the final trust region radius Δ_{end} and the max number of iteration MAX_{iter}. Given these three parameters, and a initial point $\mathbf{x}_0 \in R^n$, the QP_TR trust region method can find the local minimum of $f\left(\mathbf{x}\right)$ around $\mathbf{x}_0$. More information can be found in ref [11].

Self-Calibration with Two Views Using the Scale-Invariant Feature Transform

Jae-Ho Yun[1] and Rae-Hong Park[1,2]

[1] Department of Electronic Engineering, Sogang University
C.P.O. Box 1142, Seoul 100-611, Korea
{yastoil, rhpark}@sogang.ac.kr
[2] Interdisciplinary Program of Integrated Biotechnology, Sogang University

Abstract. In this paper, we present a self-calibration strategy to estimate camera intrinsic and extrinsic parameters using the scale-invariant feature transform (SIFT). The accuracy of the estimated parameters depends on how reliably a set of image correspondences is established. The SIFT employed in the self-calibration algorithms plays an important role in accurate estimation of camera parameters, because of its robustness to changes in viewing conditions. Under the assumption that the camera intrinsic parameters are constant, experimental results show that the SIFT-based approach using two images yields more competitive results than the existing Harris corner detector-based approach using two images.

Keywords: Self-Calibration, Scale-Invariant Feature Transform, Harris Corner Detector.

1 Introduction

Camera calibration is the process of determining the camera intrinsic and extrinsic parameters that represent a camera's geometry. Most existing methods use a calibration object, such as a checkboard pattern, whose geometry in the three-dimensional (3-D) space is known [1]. In contrast, camera self-calibration does not need any calibration object. Self-calibration is performed with correspondences between several views of the same scene instead of using a calibration object. Many self-calibration methods based on the fundamental matrix have been proposed [2][3]. These methods at first compute the fundamental matrix from the corresponding points of the image. Then, intrinsic parameters and the relative pose of a camera are retrieved up to a projective transformation in the 3-D space [4]. However, most self-calibration methods are sensitive to noise. Shift in correspondences by just one pixel in pixel coordinates can produce a wrong estimate of camera parameters. Thus, these methods use lots of image sequences to decrease the effect of image noise. Despite of that, they often fail to get the accurate results.

This paper shows the superiority of the distinctive image features, scale-invariant feature transform (SIFT) that is invariant to image scaling and rotation [5], in the self-calibration framework. It provides us with the robustness against changes in 3-D viewpoint and decreases the probability of the incorrect estimation of parameters, giving reliable image correspondences. Thus, employment of the SIFT into the self-calibration framework can improve the estimation accuracy of the calibration parameters. The improvement is measured compared with the existing Harris corner detector [6].

The rest of the paper is structured as follows. Section 2 reviews two-view geometry, camera model, and epipolar geometry. Section 3 explains the SIFT-based image correspondences. Section 4 describes three self-calibration methods. Section 5 gives experimental results and discussions. Section 6 concludes the paper.

2 Two-View Geometry

The theoretical bases for the two-view geometry are presented in this section. We begin with the general pinhole camera model that describes a mapping between the 3-D space and a two-dimensional (2-D) image. Then, we will review the epipolar geometry and the fundamental matrix defined between two views.

2.1 Camera Model

Under the pinhole camera model, a point $\mathbf{X} = (x, y, z, 1)^T$ in the 3-D space is mapped to the point $\mathbf{x} = (u, v, 1)^T$ in the 2-D image plane, in which the homogeneous coordinates are used. This mapping relationship may be written in vector-matrix form as

$$\mathbf{x} = A[R \mid t]\mathbf{X} \tag{1}$$

where A represents a camera calibration matrix, R and t denote the rotation matrix and translation vector, respectively. This equation is defined up to an arbitrary scale. The camera calibration matrix A can be written as

$$A = \begin{bmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

where α_u and α_v represent the focal lengths of the camera in terms of pixel dimensions in the u and v directions, respectively. (u_0, v_0) signifies the image coordinates of the principal point and s denotes the skew parameter.

2.2 Epipolar Geometry

The epipolar geometry describes the fundamental geometric relationship between two perspective cameras. It depends on the camera's intrinsic parameters and relative pose. The epipolar geometry is represented by a 3×3 matrix called the fundamental matrix F . Given a pair of images, the fundamental matrix describes the relationship between points in one image and a corresponding epipolar line in the other image. The fundamental matrix can be computed from a set of correspondences. If points $\mathbf{x}$ and $\mathbf{x}'$ are corresponding points in the 2-D image planes, the fundamental matrix satisfies the equation

$$\mathbf{x}'F\mathbf{x} = 0. \tag{3}$$

When there are more than eight correspondences, the fundamental matrix can be determined by a linear least squares minimization method [7].

3 SIFT-Based Image Correspondences

Most self-calibration algorithms estimate camera calibration parameters based on the fundamental matrix. The accurately estimated fundamental matrix can give the accurate calibration parameters. In addition, it is obvious that the accuracy of the fundamental matrix depends on the robustness of image correspondences. Existing algorithms have used the Harris corner detector to extract image feature points and developed the point matching method through evaluating the correlation between feature points [6]. However, the Harris corner detector has a limited ability to find lots of reliable image correspondences under the 3-D view change conditions.

Lowe proposed a new method for extracting distinctive image features and finding a reliable matching called the SIFT [5]. These features provide invariance under the conditions of the image scaling, rotation, and partial 3-D view changes. According to [5], features are detected with the following four steps: detection of scale-space extrema, accurate keypoint localization, orientation assignment, and descriptor generation. First, the difference-of-Gaussian (DoG) spaces are generated by convolving the original image with the Gaussian filter mask having a varying scale parameter, producing a set of Gaussian images, and subtracting them to construct a set of DoG images. The maxima and minima of the DoG images are selected as feature candidates. Second, their stability is measured with the ratio of principal curvatures. If the ratio is below some threshold, the selected feature is discarded. After that, the interpolated location of each feature is determined by fitting feature points to a 3-D quadratic function. Third, the reference direction is assigned to each feature based on the largest image gradient direction in the specified region. Fourth, the image gradients in the neighboring region of the feature are measured to specify the characteristics of the local region. The image gradients described above are represented by a local image descriptor vector. This approach can generate a large number of stable features.

The reliable correspondences are established by individually comparing each feature of images. If two features have the minimum Euclidean distance between their image descriptor vectors, this feature pair could be a best match. As a result, the image correspondences established using the SIFT are more reliable than the correspondences by the Harris corner detector-based method. Furthermore, the superiority of the SIFT has been demonstrated by a recent paper [8]. The SIFT-based self-calibration approach can improve the estimation accuracy of the calibration parameters, which was also presented in [9].

4 Self-Calibration

Self-calibration is a computational work to estimate the camera intrinsic parameters, such as focal lengths, the image center, and the camera skew, from a set of uncalibrated images. This section describes three methods for determination of camera intrinsic parameters, which are all based on estimation of the fundamental matrix. Next, camera extrinsic parameters are estimated. Then, the camera projection matrix can be determined with the estimated camera intrinsic and extrinsic parameters.

4.1 Intrinsic Parameter Estimation

There are various self-calibration algorithms to estimate the camera intrinsic parameters. This paper focuses on three self-calibration algorithms that were recently presented. Among them, two algorithms use the Kruppa's equations whereas the other is based on the eigenvalues of the essential matrix.

Whitehead and Roth suggested the estimation algorithm which starts with the singular value decomposition (SVD) of the fundamental matrix [2]. Let the SVD of the fundamental matrix F be UDV^T, A be the camera calibration matrix, and K be the symmetric matrix expressed as AA^T. The Kruppa's equation is given by

$$\frac{\mathbf{v}_2^T K \mathbf{v}_2}{r^2 \mathbf{u}_1^T K \mathbf{u}_1} = \frac{-\mathbf{v}_1^T K \mathbf{v}_2}{sr \mathbf{u}_2^T K \mathbf{u}_1} = \frac{\mathbf{v}_1^T K \mathbf{v}_1}{s^2 \mathbf{u}_2^T K \mathbf{u}_2} \tag{4}$$

where $\mathbf{u}_1$, $\mathbf{u}_2$, and $\mathbf{u}_3$ represent column vectors of U whereas $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$ denote column vectors of V. r and s are the singular values from $D = diag(r, s, 0)$. Given N fundamental matrices, the intrinsic parameters can be estimated using the non-linear least squares problem written as

$$\sum_{i=1}^{N} w_i \left(factor_1^2 + factor_2^2 + factor_3^2 \right) \tag{5}$$

where w_i is a weight factor, with $factor_1$ equal to

$$\frac{\mathbf{v}_2^T K \mathbf{v}_2}{r^2 \mathbf{u}_1^T K \mathbf{u}_1} - \frac{-\mathbf{v}_1^T K \mathbf{v}_2}{sr \mathbf{u}_2^T K \mathbf{u}_1} \tag{6}$$

and $factor_2$ and $factor_3$ can be defined similarly.

Mendonça and Cipolla presented another method based on the essential matrix [10]. The essential matrix is given by

$$E = A^T F A \tag{7}$$

where A represents the camera calibration matrix. When σ_1 and σ_2 are the two singular values of E, where $\sigma_1 > \sigma_2$, and N essential matrices are given, the intrinsic parameters can be estimated using the non-linear least squares problem that can be written as

$$\sum_{i=1}^{N} w_i \left(1 - \frac{\sigma_2}{\sigma_1} \right) \tag{8}$$

where w_i is a weight factor.

Lastly, Habed and Boufama proposed a simplified form of the Kruppa's equation [11], which is given by

$$FKF^T KF = F \tag{9}$$

where F represents the fundamental matrix. Also Habed and Boufama defined the cost function to refine the solutions of equation (9) using the nonlinear least-square method. The cost function is written as

$$\left\| \frac{FKF^T KF}{\left\| FKF^T KF \right\|_F} - \frac{F}{\left\| F \right\|_F} \right\|_F \tag{10}$$

where $\| \, . \, \|_F$ denotes the Frobenius norm.

4.2 Extrinsic Parameter Estimation

After the intrinsic parameters are derived, determination of the camera extrinsic parameters is needed to estimate two projection matrices. In the case of the known intrinsic parameters, the relative pose of the camera can be derived from the essential matrix defined by two views. With the fundamental matrix and the calibration matrix, the essential matrix is computed by equation (7). On the other hand, the essential matrix has the another form by definition

$$E = [t]_\times R \tag{11}$$

where $[t]_\times$ represents a 3×3 skew-symmetric matrix of the translation vector and R denotes the rotation matrix.

Hartley proposed the estimation algorithm of the relative camera position [4]. The rotation matrix and the translation vector can be estimated, up to a scale factor, by the factorization of the essential matrix using the SVD.

5 Experimental Results and Discussions

Most self-calibration algorithms evaluate their performance with virtually generated points in the 3-D space as well as real images. Because the performance by the SIFT can be estimated with real images, we take only real image pairs to perform the self-calibration. We estimate the camera intrinsic parameters with two kinds of real image pairs that have been widely used to measure the performance of the calibration method. Those image pairs were derived from the Valbonne church and Leuven castle sequences.

The SIFT is applied to the two image pairs to establish correspondences between the two images of each pair. A demonstration version of the SIFT can be found in the website, http://www.cs.ubc.ca/~lowe/keypoints/. The ratio of distances among the SIFT parameters, that is calculated by dividing the minimum Euclidean distance between two image descriptor vectors of the correspondences by the next minimum Euclidean distance, is set to a low value of 0.3 to find robust correspondences between two views. The ratio of distances is in proportion to the probability of the false correspondences. Using the SIFT, 139 and 88 correspondences are found from the Valbonne church and Leuven castle image pairs, respectively. Next, the fundamental matrix is estimated using corresponding points. The random sample consensus (RANSAC) based fundamental matrix estimation method is used [12]. In computation of the fundamental matrix, only one correspondence is classified as an outlier in the Valbonne church image pair and none in the Leuven castle image pair, because robust correspondences using the SIFT are employed.

The Harris corner detector and correlation-based matching are also applied to the two image pairs to establish correspondences. We select the parameter values of each method in such a way that each method has approximately the same number of correspondences that are classified as inliers in estimating the fundamental matrix. In comparison with the SIFT, the Harris corner detector-based approach needs more candidate correspondences to have approximately the same number of the correspondences that are classified as inliers. Initially, 183 and 247 correspondences are found from the Valbonne church and Leuven castle image pairs, respectively, using the Harris corner detector and correlation-based matching.

However, 138 and 89 correspondences are used as inliers by the RANSAC method in estimating the fundamental matrix in the Valbonne church and Leuven castle image pairs, respectively. In the case of applying the Harris corner detector to the Leuven castle image pairs, about half of the correspondences are extracted from the tree region of the images. Because the similar image patterns are found in the tree region, it is unlikely that robust correspondences are established between the tree regions of an image pair. Then, most correspondences that are located in the tree region are classified as outliers by the RANSAC method, as expected. We try in vain to select a higher threshold value in running the Harris corner detector to remove the feature points found in the tree region. Thus, the Harris corner-based matching shows a higher outlier rate than the SIFT. Figs. 1 and 2 show the correspondences, detected by the SIFT and the Harris corner detector, of the Valbonne church and Leuven castle image pairs, respectively. White and black circles in each image pair represent the inliers and outliers that are classified by the RANSAC method, respectively. We can easily see that there are lots of black circles in Fig. 2(b), because 172 correspondences are classified as outliers, especially in the tree region.

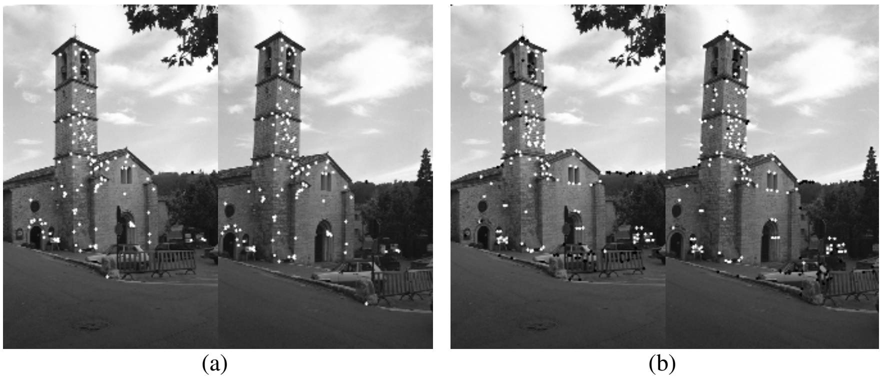
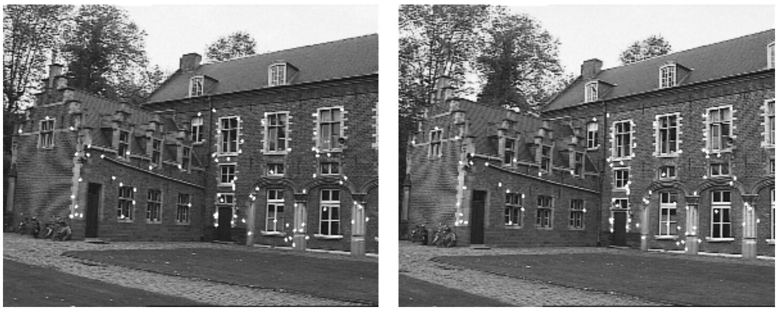


Fig. 1. Correspondences of the Valbonne church image pair. (a) Using the SIFT. (b) Using the Harris corner detector and correlation-based matching.

Before performing self-calibration using three existing algorithms described in Section 4.1, we make some assumptions to reduce the number of parameters to be estimated, because only the camera focal length could be estimated given a single fundamental matrix. We assume that the camera has a zero skew and the principal point is set to be the center of the image. In addition, the ratio α_u / α_v is assumed to be equal to unity and the focal lengths are constant. These assumptions are practically reasonable.

The resolution of the Valbonne church image is 512×768. Thus, the principal point is set at the center of the image (256, 384). Similarly, the principal point of the Leuven castle image is set at (384, 288), noting that the Leuven castle image has a size of 768×576. Three self-calibration methods are applied to each image pair using the fundamental matrices that are estimated through the SIFT-based and the Harris corner detector-based approach. The solutions of each cost function are obtained using the non-linear least squares method. Table 1 shows the comparison of the estimated focal lengths using the SIFT-based approach, those using the Harris corner detector-based approach, and the ground truths.

The estimated focal lengths of the Valbonne church image pair are similar, 692 and 691, when we use the SIFT-based method. However the estimated focal lengths of the Valbonne church image pair using the Harris corner detector-based method are far from the ground truth. The SIFT-based and Harris corner detector-based self-calibration methods give average errors of about 1.4% and 7.1%, respectively, with respect to the ground truths. It is sure that the SIFT-based approaches produce more accurate results than the Harris corner detector-based methods. When we use the Leuven image pair, the SIFT-based methods give the focal lengths that are also close to the ground truth with the error of 0.54% except for the result of Mendonça and Cipolla's method. The Mendonça and Cipolla's method with the SIFT produces the focal length that is far from the ground truth, with the error of 36%. The Harris corner detector-based methods have the errors three times larger than the SIFT-based


(a)

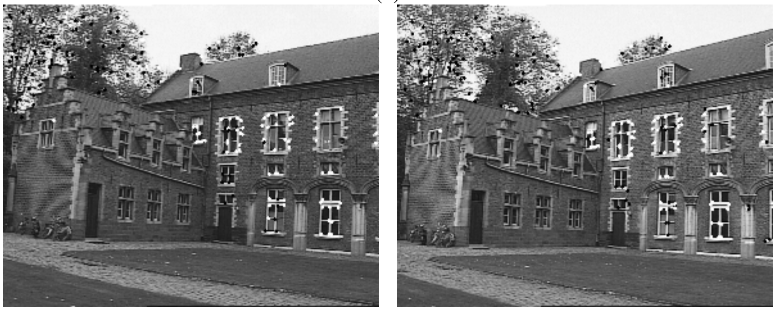

(b)

Fig. 2. Correspondences of the Leuven castle image pair. (a) Using the SIFT. (b) Using the Harris corner detector and correlation-based matching.

methods. Also the Harris corner detector-based methods show a similar trend except that the error of the estimated focal length by the Mendonça and Cipolla's method is extremely large. Thus, we can say that the SIFT-based approach can improve the estimation accuracy of the focal length compared with the Harris corner detector-based method.

With the estimated focal length and the assumption about the camera skew and the principal point described above, we can derive the complete camera calibration matrix. The rest of the camera projection matrix, a rotation matrix and translation vector, can be estimated with the camera calibration matrix and the fundamental matrix. Hartley proposed the method to estimate a relative camera pose using the factorization of the essential matrix, which is mentioned in Section 4.2. Through these processes, we can determine the 3×4 camera projection matrices corresponding to two respective views, up to a scale factor. The validity of the estimated projection matrices can be shown by reconstructing the positions of all the feature points in the 3-D space. 3-D positions of all the correspondences are determined by the triangulation method [13]. Fig. 3 shows the estimated 3-D positions of feature points that are extracted from each image pair. Fig. 3(a) has a difficulty in recognizing Valbonne church's geometry, because it has a small depth variation. In contrast, Fig. 3(b) shows the coarse structure of the Leuven castle.

Table 1. Comparison of the estimated focal lengths by three self-calibration methods and the ground truths (pixels)

Methods		Valbonne church images	Leuven castle images
Ground truths		682	1100
Whitehead and Roth's method [2]	SIFT-based	692	1094
	Harris corner detector-based	637	907
Mendonça and Cipolla's method [10]	SIFT-based	691	701
	Harris corner detector-based	628	570
Habed and Boufama's method [11]	SIFT-based	692	1094
	Harris corner detector-based	636	906

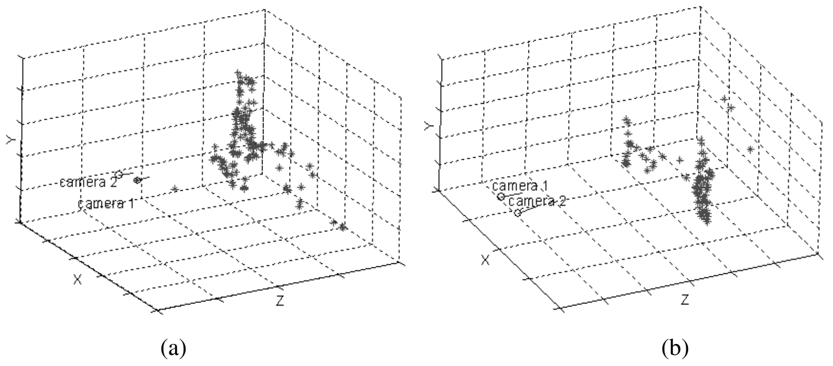


Fig. 3. 3-D reconstructions of each image pair. (a) Valbonne church. (b) Leuven castle.

6 Conclusions

This paper shows the superiority of a SIFT-based self-calibration strategy to increase the estimation accuracy of the camera intrinsic parameters. Its performance is compared with that of the Harris corner-based approach with two image pairs. The results estimated using the SIFT-based approach are closer to the ground truths than those by the Harris corner detector-based approach. As a result, it can be said that the SIFT-based approach can improve the estimation accuracy of the focal length compared with the Harris corner detector-based method. Further research will focus on the development of the automatic parameter selection method to extract a number of robust correspondences, keeping the outlier rate low.

Acknowledgement

This work was supported by Brain Korea 21 Project.

References

1. Zhang. Z.: Flexible Camera Calibration by Viewing a Plane from Unknown Orientations. Proc. Int. Conf. Computer Vision, Vol. 1. Kerkyra Corfu (1999) 666–673
2. Whitehead, A., Roth, G.: Estimating Intrinsic Camera Parameters from the Fundamental Matrix Using Evolutionary Algorithm. EURASIP J. Applied Signal Processing, 2004 8 (2004) 1113–1124
3. Lourakis, M. I. A., Deriche, R.: Camera Self-Calibration Using the Singular Value Decomposition of the Fundamental Matrix. Proc. Asian Conf. Computer Vision, Vol. 1. Taipei Taiwan (2000) 403–408
4. Hartley, R. I.: Estimation of Relative Camera Positions for Uncalibrated Cameras. Proc. European Conf. Computer Vision, Santa Margherita Ligure Italy (1992) 579–587
5. Lowe, D. G.: Distinctive Image Features from Scale-Invariant Keypoints. Int. J. Computer Vision, 62 2 (2004) 91–110
6. Harris, C., Stephens, M.: A Combined Corner and Edge Detector. Proc. Conf. Alvey Vision, Manchester UK (1988) 147–151
7. Hartley, R.: In Defence of the Eight-Point Algorithm. IEEE Trans. Pattern Analysis and Machine Intelligence, 19 6 (1997) 580–593
8. Moreels, P., Perona, P.: Evaluation of Features Detectors and Descriptors Based on 3D Objects. Proc. Int. Conf. Computer Vision, Vol. 1. Beijing China (2005) 800–807
9. Gordon, I., Lowe, D. G.: Scene Modeling, Recognition and Tracking with Invariant Image Features. Proc. Int. Sym. Mixed and Augmented Reality, Arlington VA USA (2004) 110–119
10. Mendonça, P., Cipolla, P.: A Simple Technique for Self-Calibration. Proc. Int. Conf. Computer Vision, Vol. 1. Kerkyra Greece (1999) 500–505
11. Habed, A., Boufama, B. S.: Self-Calibration of a Simplified Camera Using Kruppa Equations. Proc. Canadian Conf. Computer and Robot Vision, London Ontario Canada (2004) 446–450
12. Hartley, R. I., Zisserman, A.: Multiple View Geometry in Computer Vision. 2nd edn. Cambridge University Press, Cambridge UK (2004)
13. Hartley, R. I., Sturm, P.: Triangulation. Int. J. Computer Vision, 68 2 (1997) 146–157

Improved Face Recognition Using Extended Modular Principal Component Analysis

Changhan Park[1,2], Inho Paek[1], and Joonki Paik[1]

[1] Image Processing and Intelligent Systems Laboratory, Department of Image Engineering, Graduate School of Advanced Imaging Science, Multimedia, and Film, Chung-Ang University, 221 Huksuk-dong, Tongjak-Ku, Seoul 156-756, Korea
[2] Advanced Technology R&D Center, Samsung Thales Co., Ltd., San 14-1, Nongseo-dong, Giheung-Gu, Yongin, Gyeonggi 446-712, Korea
changhan.park@samsung.com
http://www.samsungthales.com

Abstract. In this paper, we present an improved face recognition algorithm using extended modular principal component analysis (PCA). The proposed method, when compared with a regular PCA-based algorithm, has significantly improved recognition rate with large variations in pose, lighting direction, and facial expression. The face images are divided into multiple, smaller blocks based on the Gaussian model and we use the PCA approach to these combined blocks for obtaining two eyes, nose, mouth, and glabella. Priority for merging blocks is decided by using fuzzy logic. Some of the local facial features do not vary with pose, lighting direction, and facial expression. The proposed technique is robust against these variations.

1 Introduction

Human biometric characteristic is unique, so it can hardly be duplicated [1]. Such information includes: facial expression, speech, hands, body, and gesture to name a few. Face detection and recognition techniques are proven to be more popular than other biometric features based on efficiency and convenience [2], [3]. It can also use a low-cost personal computer (PC) camera instead of expensive equipments, and requires minimal user interface. Face recognition is a difficult problem with a variety of sensitive factors such as face change with facial expression, pose, viewpoint, illumination, noise, and age.

Recently, extensive research using 3D face data has been carried out in order to overcome the limits of 2D face detection and extraction [2], which include principal component analysis (PCA) [3], artificial neural networks (ANN) [4], support vector machines (SVM) [5], hidden markov models (HMM) [6], and linear discriminant analysis (LDA) [7]. Among them, PCA and LDA methods with self-learning are most widely used [3], and at the same time combination of various methods tend to become alternative solutions. While most traditional face recognition techniques are focused on the whole face, localized parts-based face recognition is more effective under special environment. In the appearance-based method of PCA, intensity and/or intensity derived parameters such as eigenfaces are used. The modular PCA [8] approach was presented to improve the accuracy of PCA in cases of extreme change of illumination and pose.

In this paper, we propose to improve the accuracy of face recognition subject to variation of pose, lighting direction, and facial expression. The eye formation is one of the most invariant features of the face, because geometric information of two eyes can provide pose and position information. The proposed method locates two eyes by using darker blobs of the input image [9], which is normalized by 60×60. Gaussian models are established for blocks of eyes, nose, mouth, glabella from the eye formation.

Although PCA is a popular technique for recognizing a frontal face, its accuracy is significantly decreased with change in pose and/or illumination. In this paper, we propose an extended version of modular PCA, whose original version has been proposed by Asari et al. [8]. The proposed method divides the input face image into smaller blocks using the Gaussian probability models of two eyes, nose, mouth, and glabella, and applies PCA to each block. We also use fuzzy logic to obtain the optimum priority of merging.

Conventional PCA technique deal with the entire face region, hence they require long training time for large variation in pose, and facial expression. On the other hand the proposed extended modular method can significantly improve the recognition rate with reduced amount of training time.

This paper is organized as follows: Section 2 describes the PCA and the singular value decomposition (SVD) theories. Section 3 explains Gaussian model of probability density function (PDF), and presents the proposed extended modular PCA with fuzzy logic-based priority decision. Section 4 summarizes experimental results of the proposed and the conventional PCA methods. Finally, section 5 concludes the paper.

2 PCA and SVD Theories Revisited

In the process of PCA we compute covariance matrix C and its eigenvectors from the training set of images. Consider the face images in the face database to be of size $N\times N$. These images can be represented as a vector of dimension N^2. Let $\{I_1, I_2, \cdots, I_M\}$ be a set of M training face vectors. By definition, C can then be estimated as [10]

$$C=\frac{1}{M}\sum_{k=1}^{M}Y_kY_k^T, \tag{1}$$

where each face differs from the average face by the vector $Y=I-\overline{m}$. The training dataset is packed into the following matrix

$$I=[I_1, I_2, \cdots, I_M]. \tag{2}$$

The average face is defined as

$$\overline{m}=\frac{1}{M}\sum_{k=1}^{M}I_k. \tag{3}$$

Even for images of moderate size, computation of eigenvectors of C is computationally complex. By using SVD of X, the eigenvectors of XX^T can be found from eigenvectors of $X^T X$, which are much easier to obtain. More specifically, suppose the rank of X is r, $r \le N$. According to fundamental linear algebra, X has SVD as

$$X = \sum_{k=1}^{r} \sqrt{\lambda_k} u_k v_k^T , \tag{4}$$

where $\sqrt{\lambda_k}$, u_k, and v_k respectively represent singular values, left, and right singular vectors of X. u_k and v_k have the following relationship.

$$u_k = \frac{1}{\sqrt{\lambda_k}} X v_k . \tag{5}$$

Hence, we can easily find eigenface u_k after finding v_k.

3 Gaussian PDF and Extended Modular PCA

In face recognition, PCA struggles with varying pose, expression, and light. For this reason we propose an improved method to minimize special conditions occurring in traditional PCA. The proposed method locates two eyes from darker blobs, and the corresponding area is normalized to 60×60. Gaussian models are established for block regions representing eyes, nose, mouth, and glabella as a preprocessing of the extended modular PCA procedure.

3.1 Gaussian PDF

Gaussian models of blocks representing left and right eyes, nose, mouth, and glabella are generated using Gaussian PDF defined as [11]

$$p(x \mid \mu, \Sigma) \cong \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{ -\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu) \right\} , \tag{6}$$

where x represents a $D \times 1$ feature vector, μ the $D \times 1$ mean vector, and Σ the $D \times D$ covariance matrix. More specifically the mean vector and the covariance matrices are defined as $\mu = E[x]$, and $\Sigma = E[(x-\mu)(x-\mu)^T]$, respectively.

Fig. 1 shows training sets of the Gaussian model of a face representing eyes, nose, mouth, and glabella.

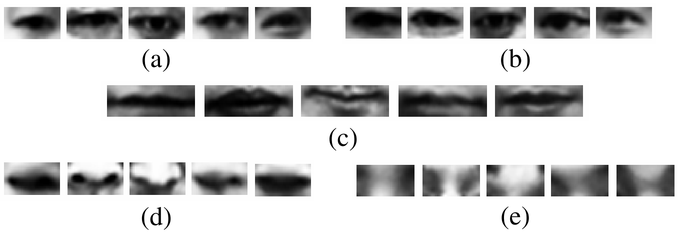


Fig. 1. Face Gaussian models of; (a) left eyes, (b) right eyes, (c) mouths, (d) noses, and (e) glabellas

3.2 Extended Modular PCA

For the extended modular PCA, the face images are divided into B smaller blocks from the previously generated Gaussian models.

Creation of new combined block using the proposed method is defined as

$$I_{new} = \sum I_{block} , \tag{7}$$

where I_{block} includes left and right eyes, nose, mouth, and glabella.

These blocks can be represented mathematically as

$$I_{ij}(m,n) = I_i\{I_{block}(j-1)+m, I_{block}(j-1)+n\}, \ \forall i, j \tag{8}$$

where i varies from 1 to M, the number of images in the training set, j varies from 1 to B, the number of blocks, and m and n respectively represent the sizes of the created image.

The average image of all the combined blocks is obtained as

$$\overline{m} = \frac{1}{M \cdot B}\sum_{i=1}^{M}\sum_{j=1}^{B} I_{ij} \ . \tag{9}$$

The next step is to normalize each training block by subtracting it from the mean as

$$\overline{X_{ij}} = I_{ij} - \overline{m}, \quad \forall_{i,j} \ . \tag{10}$$

From the normalized combined blocks the covariance matrix is computed as

$$C = \frac{1}{M \cdot B}\sum_{i=1}^{M}\sum_{j=1}^{B} \overline{X_{ij}} \cdot \overline{X_{ij}}^{T} \ . \tag{11}$$

In this method, estimating eigenvalues and eigenvectors of C is computationally complex, and requires long processing time. This problem can be solved by using SVD.

Given that $\overline{X}_{ij}$ replaces with X, SVD is a factorization of X as $X = UDV^T$, where U and V represent orthogonal matrices, and D represents a diagonal matrix with non-negative entries. Effective number k selects value larger than threshold t in D, and selects the left eigenvector as many as k in U.

The estimated eigenvector v of the selected k in U is sorted by the corresponding eigenvalue. Therefore, the eigenvector V can be expressed as a set of k eigenvectors as

$$V = [v_1, v_2 \cdots v_k] \cdot \tag{12}$$

Given the set of sorted eigenvectors, the weight for each image in the training set is computed by the following projection

$$W_{iK} = V_K^T \cdot (I_{ij} - m), \quad \forall_{i,K}, \tag{13}$$

where V_K's represent eigenvectors with k largest eigenvalues of X, as K varies from 1 to k.

3.3 Classification for Face Recognition

Recognized faces are classified based on Mahalanobis distance [11] as

$$d(x_i, x_j) \equiv (x_i - x_j)^T \Sigma^{-1} (x_i - x_j), \tag{14}$$

where x_i represents the feature value of the projected image, x_j represents the feature value of a new input image, and Σ^{-1} represents the covariance matrix of the training weights. If $\min(d) < \theta_i$, the test image is classified to the corresponding class, where θ_i represents the threshold.

3.4 Fuzzy Logic for Improving Recognition

In this subsection, we propose a recognition method to consider the merging priority based on fuzzy logic. First, the proposed method extracts a list of recognized candidate blocks. The priority of merging these blocks is decided by a fuzzy logic. The proposed method assigns *P5* to a set of 5 blocks of potential merging. Other methods assign *P1*, *P2*, *P3*, and *P4* to 1, 2, 3, and 4 blocks, respectively. We perform fuzzy inferences 1st, 2nd, 3rd, 4th, and 5th orders for minimum distance of face sub-blocks. This can drive good results of face sub-blocks. This makes *F1* occur over 1 once for the face sub-block of the same person. Other methods define *F2*, *F3*, *F4*, and *F5* by using sets of 2, 3, 4, and 5 blocks, respectively. For the proposed fuzzy inference steps we used Kim's method [12]. The corresponding fuzzy engine is classified by a recognized probability as shown in Fig. 2.

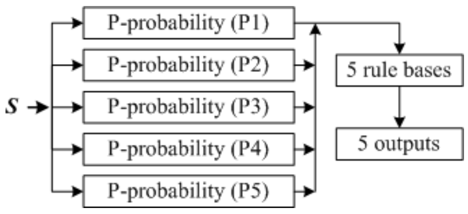


Fig. 2. Fuzzy inference engine

In Fig. 2, *P1*, *P2*, *P3*, *P4*, and *P5* represent the corresponding coefficient of recognized probability. The base rule is defined as

$$O_\theta = \begin{cases} 1.0, & \text{P(R)} = \text{ZERO} \\ 0.5, & \text{P(R)} = \text{SMALL} \\ 0.0, & \text{P(R)} = \text{LARGE} \end{cases}, \quad (15)$$

where R represents one of *F1*, *F2*, *F3*, *F4*, and *F5*.

Input membership function of the fuzzy inference engine is shown in Fig. 3.

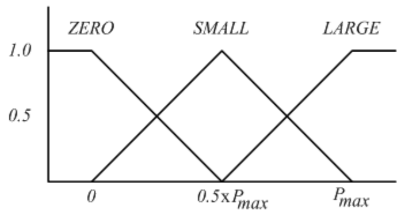


Fig. 3. Input membership function of fuzzy engine

Finally, predicted mergence can be written by using the Singleton fuzzier, the product inference engine, and the average defuzzifier as

$$P_{\max} = F1(1/O_1) + F2(1/O_2) + F3(1/O_3) + F4(1/O_4) + F5(1/O_5) \cdot \tag{16}$$

4 Experimental results

Five sets of image databases are used for the experiment. The Olivetti Research Laboratory (ORL) database consists of images which were taken at different times, with variation of lighting, facial expressions, and facial details as shown in Fig. 4. The Yale database consists of images with varying illumination and expressions as shown in Fig. 5. The University of Manchester Institute of Science and Technology (UMIST) database consists of images with varying poses as shown in Fig. 6.

Fig. 4. ORL face image set

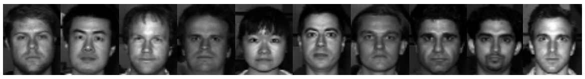


Fig. 5. Yale face image set

Fig. 6. UMIST face image set

Fig. 7 shows the result of face separation by using the multi-layered relative edge map, and the experimental face database was made by this result.

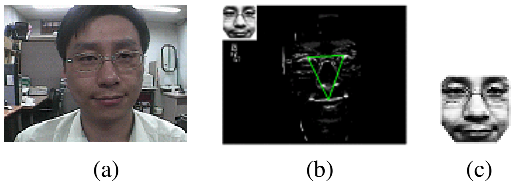


Fig. 7. (a) An input image, (b) the correspondingly created edge map, and (c) the normalized database of size 60×60

The Bio database consists of images with varying poses and expressions as shown in Fig. 8, which shows the detected face region in the Bio database by using the proposed method. Our database, called IPIS database, consists of images with varying pose and facial expressions as shown in Fig. 9. In experimental results of conventional PCA techniques, the whole face is considered, which causes a delay in processing due to a large variation in the frontal face, facial expression, and pose.

Fig. 8. Bio face image set

Fig. 9. IPIS face image set

A recognition pattern using the entire face images is called the global recognition pattern (GRP), which uses the combined facial part called the region recognition pattern (RRP). For convenience, 1, 2, 3, 4, and 5 represent left and right eyes, nose, mouth, and glabella, respectively. Fig. 10 compares experimental results of the proposed method. 400 training images were used for the experiment as a database, and 100 used test images.

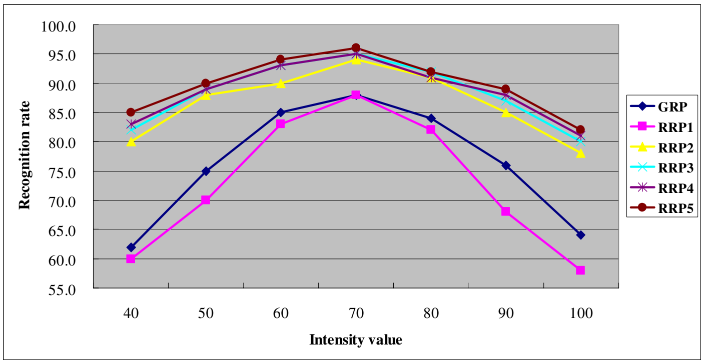


Fig. 10. Comparison of experimental result

Fig. 11 shows comparative results of training time, where the proposed method can be showed to be in real-time. Fig. 12 shows frequency by using the proposed method. In the experiment, In Yale database recognition rate falls in the special area from intensity of light. The proposed method provided significantly improved recognition rate.

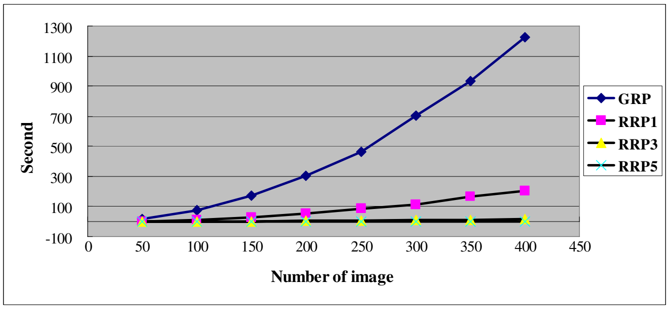


Fig. 11. Comparative results of training time

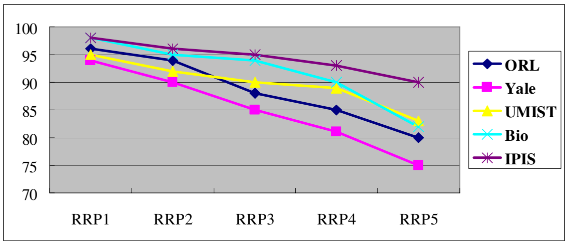


Fig. 12. Result of frequency by each block

5 Conclusion

In this paper, we presented a face recognition algorithm based on extended modular PCA approach. The proposed method, when compared with conventional PCA-based algorithms, significantly improved the recognition rate for face images with large variations of pose, lighting direction, and facial expression. In the proposed method, the face images are divided into smaller blocks with Gaussian model, and PCA is applied to each block representing two eyes, nose, mouth, and glabella. Since some local facial features of an individual do not vary even when the pose, lighting direction and facial expression, we expect the proposed technique can reduce large variations. The accuracies of the conventional PCA-based methods and the proposed extended modular PCA method are evaluated and compared under the same conditions of varying pose, expression, and illumination using several standard face databases. Based on experimental results the proposed method can be recognize faces in real-time with significantly improved recognition accuracy.

Acknowledgment

This research was supported by Korean Ministry of Science and Technology under the National Research Laboratory Project, Korean Ministry of Information and Communication under HNRC-ITRC program at Chung-Ang University supervised by IITA, and by Korean Ministry of Education under the BK21 Project. The authors thank David T. Paik for his English proof-reading.

References

1. S. Kong, J. Heo, B. Abidi, J. Paik, and M. Abidi, "Recent advances in visual and infrared face recognition - A review," *Computer Vision, Image Understanding*, vol. 97, no. 1, pp. 103-135, January 2005.
2. M. Yang, D. Kriegman, and N. Ahuja, “Detecting faces in images: a survey,” *IEEE Trans. Pattern Analysis, Machine Intelligence*, vol. 24, no. 1, pp. 34-58, January 2002.
3. Z. Sun, G. Bebis, X. Yuan, and S. Louis, “Genetic feature subset selection for gender classification: a comparison study,” *Proc. 2002 Sixth IEEE Workshop, Applications of Computer Vision*, pp. 165-170. December 2002.

4. H. Rowley, S. Baluja, and T. Kanade, “Neural Network-based face detection,” *IEEE Trans. Pattern Analysis, Machine Intelligence*, vol. 20, no. 1, pp. 203-208, January 1998.
5. E. Osuna, R. Freund, and F. Girosi, “Training support vector machines: an application to face detection,” *Proc. 1997 IEEE, Computer Vision, Pattern Recognition*, pp. 130-136, June 1997.
6. F. Samaria and S. Young, “HMM based architecture for face identification,” *Image, Vision Computing*, vol. 12, no. 8, pp. 537-543, October 1994.
7. P. Belhumeur, J. Hespanha, and D. Kriegman, “Eigenfaces vs fisherfaces: recognition using class specification linear projection,” *IEEE Trans. Pattern Analysis, Machine Intelligence*, vol. 19, no. 7, pp. 711-720, July 1997.
8. R. Gottumukkal and V. Asari, “An improved face recognition technique based on modular PCA approach,” *Pattern Recognition Letters*, vol. 25, pp. 429-436, 2004.
9. Y. Kim, C. Park, and J. Paik, “A new 3D active camera system for robust face recognition by correcting pose variation,” *Proc. 2004 Int. Conf. Circuits, Systems*, pp. 1482-1487, August 2004.
10. J. Zhang, Y. Yan, and M, Lades, "Face recognition: eigenface, elastic matching, and neural nets," *Proc. IEEE*, vol. 85, no. 9, pp. 1423-1435, September 1997.
11. R. Duda, P. Hart, and D. Stork, *Pattern Classification*, Second Edition, John Wiley & Sons, 2001.

Shape Reconstruction by Line Voting in Discrete Space

Kosuke Sato[1], Atsushi Imiya[2], and Tomoya Sakai[2]

[1] School of Science and Technology, Chiba University, Japan
ksato@graduate.chiba-u.jp
[2] Institute of Media and Information Technology, Chiba University, Japan
Yayoi-cho 1-33, Inage-ku, Chiba, Japan, 263-8522
{imiya, tsakai}@faculty.chiba-u.jp

Abstract. Shape from silhouettes is a binary geometric tomography since both objects and projections, which are measured as silhouettes, are binary. In this paper, we formulate shape from silhouettes in the three-dimensional discrete space. This treatment of the problem implies an ambiguity theorem for the reconstruction of objects in discrete space. Furthermore, we show that in three-dimensional space, it is possible to reconstruct a class of non-convex objects from a collection of silhouettes though on a plane non-convex object is unreconstractable from any collection of silhouettes.

1 Introduction

The reconstruction of three-dimensional shapes from measured data such as range data, photometric information, and stereo image pairs, is called “Shape from X.” In this paper, we deal with “Shape from silhouettes.” This problem is also called ‘Shape from counter,” [2] and “Shape from profile” [7] in computer vision and “Shape from plane probing,” [3], in computational geometry. In computer vision, theoretical analysis of reconstruction algorithms is paid little attention.

This paper aims to introduce a complete discrete version of “Shape from silhouettes.” In this paper, a discrete object is reconstructed from discrete silhouettes, that is, both a camera which observes silhouettes and detectors which measure silhouettes are described as a voxel and a collection of voxels in the three-dimensional discrete space. A discrete object is reconstructed by line voting [15,6] in the discrete space. Therefore, a discrete object is reconstructed as the common set of a collection of discrete lines which are defined by camera voxels and silhouette voxels in the discrete space.

The illumination problem [5] estimates the minimum and maximum numbers of view points for the reconstruction of a convex body from their views from an appropriate set of view points. The illumination problem is equivalent to shape reconstruction from silhouettes or shadows. However, it is in general difficult to answer the configuration of view points for a given object. There are many results for the reconstruction of a convex polygon from their shadows. For example see [12], and [9]. Laurentini [10,11] was concerned with geometric properties of silhouette-based shape reconstruction for polyhedrons, and clarified the relation among the visible hull and the convex hull of a polyhedron.

It is possible to decompose a three-dimensional objects to a collection of shadows. The geometric relation permits to decompose shadows of a three-dimensional object to shadows of planar objects. Using this geometric relations, we reconstruct non-convex objects is reconstructible from a series of shadows.

Shape reconstruction from silhouette is a conventional technique for the detection of shape models and shape reconstruction in computer graphics, computer vision [6], and robotics [7,8,12,14]. Shape reconstruction from silhouette is achieved by visible voting[15], shape carving[6] and so on. Though these reconstruction methods are mathematically formulated in the continuous framework [13,18], the reconstruction is achieved in discrete space. In this paper, we deal with the shape-from-silhouette problem in the discrete space $\mathbf{Z}^3$.

2 Shape and Shadow

In n-dimensional Euclidean space $\mathbf{R}^n$ for $n \geq 2$, let $\boldsymbol{\omega}$ be the unit vector on S^{n-1}. A finite closed convex body K in $\mathbf{R}^n$ is expressed as

$$K = \{\boldsymbol{x}|\boldsymbol{x}^\top\boldsymbol{\omega} \leq p(\boldsymbol{\omega}),\, \boldsymbol{\omega} \in S^{n-1}\}, \tag{1}$$

where $p(\boldsymbol{\omega})$ is the distance from the origin to a tangent plane to K [1]. An intersection of a plane $\boldsymbol{\sigma}^\perp$, which is perpendicular to the unit vector $\boldsymbol{\sigma}$ and passes through the origin, and finite closed convex body K is a shadow of K projected from the direction $\boldsymbol{\sigma}$, that is,

$$S(\boldsymbol{\sigma}) = \{\boldsymbol{x}|\boldsymbol{x}^\top\boldsymbol{\sigma} = 0\} \cap K. \tag{2}$$

Let $\partial S(\boldsymbol{\sigma})$ be the boundary curve of the shadow on plane $\boldsymbol{\sigma}^\perp$. Setting $\boldsymbol{x}^\top\boldsymbol{\omega} = p(\boldsymbol{\omega})$ to be the tangent plane to $\partial S(\boldsymbol{\sigma})$, it is possible to reconstruct a finite closed convex body K from shadows observed from all directions on S^{n-1} [4].

For a point $\boldsymbol{a}$ in $\mathbf{R}^n$, a line in $\mathbf{R}^n$ is

$$L(\boldsymbol{a}, \boldsymbol{\omega}) = \{\boldsymbol{x}|\boldsymbol{x} = \boldsymbol{a} + t\boldsymbol{\omega},\, t \in \mathbf{R}\}. \tag{3}$$

For a finite closed convex body K,

$$C(\boldsymbol{a}) = \{\boldsymbol{\omega}|K \cap L(\boldsymbol{a}, \boldsymbol{\omega}) \neq \emptyset\}, \tag{4}$$

is the view cone with respect to the view point $\boldsymbol{a}$. A set

$$P(\boldsymbol{a}) = C(\boldsymbol{a}) \cap \boldsymbol{a}^\perp, \tag{5}$$

where $\boldsymbol{a}^\perp$ is the plane which is perpendicular to vector $\boldsymbol{a}$ and passes through the origin, is a shadow observed by the perspective projection.

For a finite convex object K in $\mathbf{R}^3$, if we can detect all planes which intersect with K, we can obtain all rays which path through K as the intersections of pairs of planes. These rays defines silhouettes. Therefore, this geometrical property implies that we can reconstruct a finite convex object in three-dimensional Euclidean space from the collection of all planes which intersect with this object.

For a finite closed region $\mathbf{O}$ we define the silhouette from a source $\boldsymbol{s} \in \mathbf{R}^n$ as

$$\Omega(\boldsymbol{s}) = \{\omega | l(\boldsymbol{s}) \cap O \neq \emptyset\} \tag{6}$$

for the half line

$$l(\boldsymbol{s}) = \{\boldsymbol{x} | \boldsymbol{x} = \boldsymbol{s} + t\omega, \omega \in S^{n-1}, t \geq 0\}. \tag{7}$$

The cross section of cone $\boldsymbol{l}(\boldsymbol{s})$, $\boldsymbol{s} \in \Omega(\boldsymbol{s})$ with hyperplane $\boldsymbol{s}^\top \boldsymbol{x} = d$ is geometrically defined as the silhouette from source $\boldsymbol{s}$. In this paper, we define the direction of the rays which yield silhouettes as the silhouettes.

If for all $\boldsymbol{s}$ in $\boldsymbol{R}^n \setminus K$, where K is a finite convex region in $\mathbf{R}^n$, $\Omega(\boldsymbol{s})$ are measured, we can reconstruct K as

$$K = \bigcap_{\boldsymbol{s} \in \mathbf{R}^n \setminus K} \left(\bigcap_{\omega \in \Omega(\boldsymbol{s})} \{\boldsymbol{x} | \boldsymbol{x} = \boldsymbol{s} + t\omega\} \right). \tag{8}$$

3 Reconstruction of Non-convex Object

In this section, we summarise the results in reference [19] on the reconstruction of non-convex object from silhouettes in $\mathbf{R}^3$.

Lemma 1. *From the collection of silhouettes which observed from vertices which lie on a sphere encircling this object, we can obtain the collection of 2-dimensional perspective projections of a slice form a point which moves on a circle encircling this object.*

For any points on the boundary, if there exists at least one unique convex slice curve which contains this point, we call this object a slice convex object. A convex closed object is slice convex. This geometric property and Lemma 1 derives the following theorem.

Theorem 1. *A slice convex object is uniquely reconstructible from the collection of silhouettes observed from vertices which lie on the whole sphere encircling this object.*

This theorem permits us for the reconstruction of a class of non-convex objects from silhouettes. Furthermore, in this expression, the axis for the reconstruction is not required to be a straight line.

For a slice convex object $\boldsymbol{V}$ with respect to axis $\lambda \boldsymbol{v}_0$ for $|\boldsymbol{v}_0| = 1$ and $\lambda \neq 0$, setting $\boldsymbol{A}[\boldsymbol{v}]$ to be a reconstructed object with respect to the axis $\lambda \boldsymbol{v}$, for $\lambda \neq 0$, we have the following theorem

Theorem 2. *For an object $\boldsymbol{V}$ the relation*

$$\boldsymbol{V} = \bigcap_{\exists \boldsymbol{v}_o \in \mathbf{S}^2} \boldsymbol{A}[\boldsymbol{v}_0] \tag{9}$$

is satisfied if $\boldsymbol{V}$ is slice convex with respect to axis $\lambda \boldsymbol{v}_0$.

If an object is defined as the common region of a finite number of slice convex objects, that is, object $\boldsymbol{V}$ is expressed as

$$\boldsymbol{V} = \bigcap_{\alpha=1}^{n} \boldsymbol{A}[\boldsymbol{a}_\alpha], \ |\boldsymbol{a}_\alpha| = 1, \tag{10}$$

for $\lambda \neq 0$, where $\lambda \boldsymbol{a}_\alpha$ is the axis with respect to which slices of an object is convex, we have the relation

$$\boldsymbol{V} = \bigcap_{\alpha=1}^{n} \boldsymbol{A}[\boldsymbol{a}_\alpha] \supseteq \bigcap_{\boldsymbol{v}\in S^2} \boldsymbol{A}[\lambda \boldsymbol{v}] \supseteq \boldsymbol{V}. \tag{11}$$

This relation leads to the following theorem.

Theorem 3. *Object* $\boldsymbol{V}$ *is reconstructed as*

$$\boldsymbol{V} = \bigcap_{\boldsymbol{v}\in S^2} \boldsymbol{A}[\boldsymbol{v}]. \tag{12}$$

These theorems show that it is possible to reconstruct a slice convex object from silhouettes using the equation

$$\mathbf{O} = \bigcap_{\boldsymbol{s}\in A} l(\boldsymbol{s}, \mathbf{O}), \tag{13}$$

where A is a closed convex manifold encircles an object $\mathbf{O}$ and $l(\boldsymbol{s}, \mathbf{O})$ is a line which passes through $\boldsymbol{s}$ and satisfies the property

$$l(\boldsymbol{s}, \mathbf{O}) \bigcap \mathbf{O} \neq \emptyset, \tag{14}$$

if we do not detect the axe of a slice convex object. Furthermore, if we can pre-detect axes of slice convex object, we can reconstruct a three-dimensional non-convex object. This property show the difference between shape from silhouettes in 2D and 3D, since, in 2D, the collection of silhouettes does not allow us the reconstruction of non-convex objects. Figure 1 shows geometrical relations of a silhouette in a space and slice-silhouettes in a space yielded from 3D silhouettes.

4 Voting Method

Setting the characteristic function in the view cone to be

$$c(\boldsymbol{x}; \boldsymbol{a}, \boldsymbol{\omega}) = \begin{cases} 1, \boldsymbol{x} \in C(\boldsymbol{a}, \boldsymbol{\omega}) \\ 0, \text{otherwise}, \end{cases} \tag{15}$$

if we vote $c(\boldsymbol{x}; \boldsymbol{a}, \boldsymbol{\omega})$ in to the space, we have a function

$$k(\boldsymbol{x}) = \sum_{\boldsymbol{a}\in \boldsymbol{A}} c(\boldsymbol{x}; \boldsymbol{a}, \boldsymbol{\omega}). \tag{16}$$

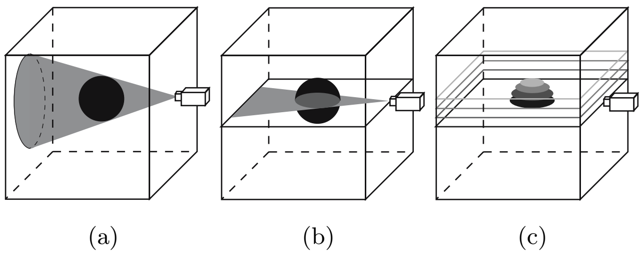


Fig. 1. Reconstruction of Object in 3D using 2D method. (a) A 3D silhouette of an object (b) A 2D silhouette as a slice of a 3D silhouette. (c) A collection of 2D slices reconstructs 3D object.

as the results of voting, where the apex of the view cone move in the region $\boldsymbol{A}$. For a positive integer τ, a set of points

$$K_\tau = \{\boldsymbol{x}|, k(\boldsymbol{x}) \geq \tau\} \tag{17}$$

defines an object. The construction of shape by K_τ is called shape reconstruction by voting. Furthermore, an algorithm for the computation of K_τ is called shape carving. Equation (16) is a geometric version of back-projection in image reconstruction from projections [16,17].

The voting process is the same operation with shape from shadows for slice convex objects, if we can obtain tangent lines at each point from all directions. Therefore, we have the following theorem.

Theorem 4. *Voting process reconstruct slice convex objects from shadows if we have orthogonal views from all directions in S^2.*

The boolean version [15] of eq. (16) for lines with finite width, which is adopted in applications is

$$K = \bigcap_{\boldsymbol{a}\in\boldsymbol{A}} \{ \bigcap_{\boldsymbol{l}(\boldsymbol{a})\in C(\boldsymbol{a},\boldsymbol{\omega})} \boldsymbol{l}(\boldsymbol{a}) \oplus \boldsymbol{B}\}, \tag{18}$$

where $\boldsymbol{B}$ is the ball with radius δ, since $\{\boldsymbol{l}(\boldsymbol{a}) \oplus \boldsymbol{B}\}$ defines a straight bar in a space whose centre line is $\boldsymbol{l}(\boldsymbol{a})$.

5 Shape Reconstruction in Discrete Space

On the discrete plane $\mathbf{Z}^2$, supercover $l(a, b, \mu)$, such that $\gcd(a, b) = 1$, is defined as

$$\{(x, y)||\boldsymbol{a}^\top \boldsymbol{x} + \mu| \leq \frac{1}{2}|\boldsymbol{a}|_1\}, \tag{19}$$

where for $\boldsymbol{x} = (x, y)^\top$ and $\boldsymbol{a} = (a, b)^\top$, $|\boldsymbol{a}|_1 = (|a| + |b|)$

In $\mathbf{Z}^3$, we adopt supercover

$$\begin{aligned}
|ax+bz+\mu_1| &\le \frac{1}{2}(|a|+|b|),\\
|ay+cz+\mu_2| &\le \frac{1}{2}(|a|+|c|),\\
|cx-by+\mu_3| &\le \frac{1}{2}(|b|+|c|),
\end{aligned} \tag{20}$$

of line in $\mathbf{R}^3$

$$\begin{pmatrix} a & 0 & b \\ 0 & a & c \\ c & -b & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \end{pmatrix}. \tag{21}$$

In $\mathbf{Z}^3$, we assume that detectors are voxels on a cube D^3 whose vertices are $(0,0,0)^\top$, $(3n-1,0,0)^\top$, $(3n-1,3n-1,0)^\top$, $(0,3n-1,0)^\top$, $(0,0,3n)^\top$, $(3n-1,0,3n-1)^\top$, $(3n-1,3n-1,3n-1)^\top$, and $(0,3n-1,3n-1)^\top$, and an object exits in a cubic region R whose vertices are $(n-1,n-1,n-1)^\top$, $(n-1,2n-1,n-1)^\top$, $(2n-1,2n-1,n-1)^\top$, $(n-1,2n-1,n-1)^\top$, $(n-1,n-1,2n-1)^\top$, $(n-1,2n-1,2n-1)^\top$, $(2n-1,2n-1,2n-1)^\top$, and $(n-1,2n-1,2n-1)^\top$. Source $\boldsymbol{s}$ moves on the faces F of D^3. We assume that our object is 6-connected simple object. Figures 2 and 3 show a discrete model of shape from silhouettes in a space.

For an object $\mathbf{O}$, Setting $\boldsymbol{l}(\boldsymbol{s},\mathbf{O})$ to be a line which passes through a fixed source $\boldsymbol{s}$, we define a collection of voxels $\boldsymbol{d}$ on D^3 such that

$$\boldsymbol{S} = \{\boldsymbol{x}|\boldsymbol{l}(\boldsymbol{s},\mathbf{O})\bigcap\mathbf{O} \neq \emptyset\}. \tag{22}$$

A collection of voxels $\boldsymbol{d}$ is the silhouette of object $\mathbf{O}$ with respect to the source $\boldsymbol{s}$.

The boundary voxels on the detector is computed as

$$\partial\boldsymbol{S} = \{\boldsymbol{S} \setminus (\boldsymbol{S} \ominus N_{26})\} \cap D^3, \tag{23}$$

where $\ominus$ expresses the Minkwoski subtraction operation.

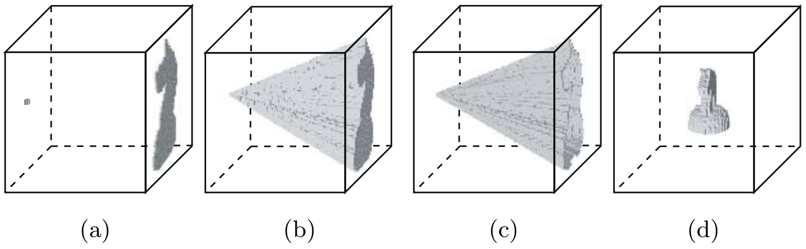


Fig. 2. Reconstruction of Object in 3D. (a) A source and a silhouette are given. (b) From a source to points on a silhouette lines are drown. (c) These lines are voted in a space. (d) The intersection of all lines is the estimation of the original object.

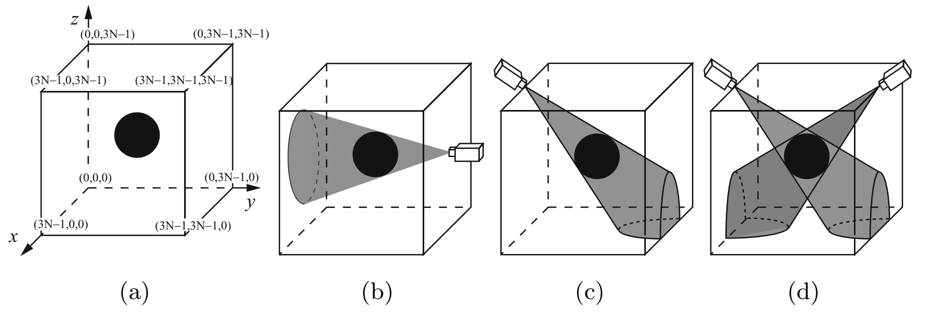


Fig. 3. Silhouette and Reconstruction of Object in 3D. (a) Geometry of the detectors and the source of 3Problem. (b) Silhouette of a convex object in a space. (c) Visible hull of a silhouette. (d) Visible hull of silhouettes.

In D^3, for voxels on a line $\boldsymbol{l}(\boldsymbol{s}, \boldsymbol{d})$ with respect to a source $\boldsymbol{s}$, we affix the labels as

$$h(\boldsymbol{s}, \boldsymbol{x}) = \begin{cases} 1, \text{if } \boldsymbol{x} \in \boldsymbol{S} \setminus \partial \boldsymbol{S} \\ 2, \text{if } \boldsymbol{x} \in \partial \boldsymbol{S} \\ 3, \text{otherwise.} \end{cases} \tag{24}$$

For these labels, we apply the operation

$$f(\boldsymbol{p}) = \max_{\boldsymbol{s} \in F}(h(\boldsymbol{s}, \boldsymbol{d})). \tag{25}$$

This operation classifies the voxels in D^3 as

$$\hat{R} = \{\boldsymbol{p} | h(\boldsymbol{p}) = 1, 2\},\ \partial \hat{R} = \{\boldsymbol{p} | h(\boldsymbol{p}) = 2\},\ D^3 \setminus \hat{R} = \{\boldsymbol{p} | h(\boldsymbol{p}) = 3\}. \tag{26}$$

This is a discrete version of eq. (18).

For these voxels, we have the next theorem.

Theorem 5. *If K is the discretisation of a finite convex region in $\mathbf{R}^3$, $\hat{K}$, which is computed using the relations in eq. (26), satisfies the relation*

$$\hat{K} \setminus K \subset (K \oplus N_{26} \oplus N_{26}) \setminus K, \tag{27}$$

where N_{26} is the eight-neighbourhood of the origin.

Proof. Consider boundary pixels whose centres are $(p, q, r)^\top$, $(p, q, r+1)^\top$, $(p+1, q, r)^\top$ and $(p, q+1, r)^\top$ of a convex object K. A supercover of a Euclidean line which touches to K is contains a bubble pixels centred at $(p, q, r+1)^\top$, $(p, q+1, r+1)^\top$, $(p+1, q+1, r+1)^\top$, $(p+1, q, r+1)^\top$, $(p, q, r+2)^\top$, $(p, q+1, r+2)^\top$, $(p+1, q+1, r+2)^\top$, and $(p+1, q, r+2)^\top$. The maximum 26-connected distance to these point from K is 2. This geometrical property proves the theorem. (Q.E.D.)

As the two-dimensional analogous of theorem 5, we have the next theorem, since slices of convex objects are convex.

Theorem 6. *If K is the discretisation of a finite convex region in $\mathbf{R}^2$, $\hat{K}$, which is computed using the relations in eq. (26), satisfies the relation*

$$\hat{K} \setminus K \subset (K \oplus N_8 \oplus N_8) \setminus K, \tag{28}$$

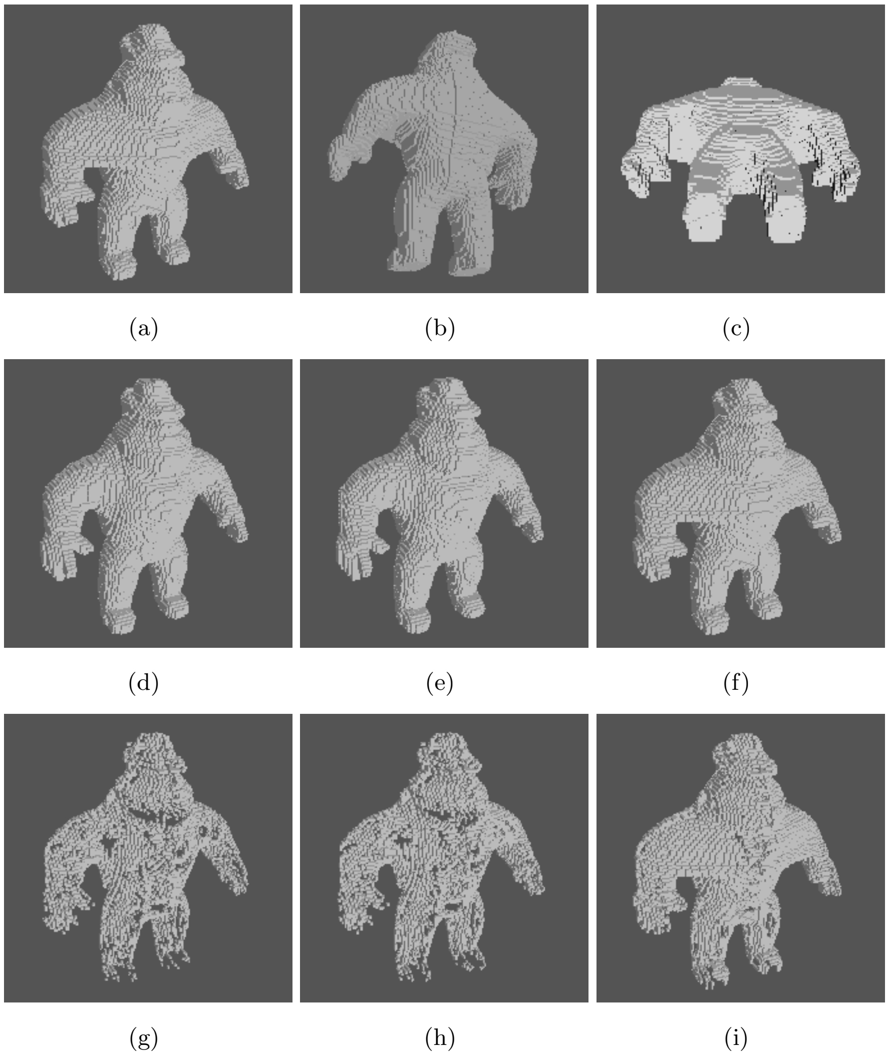


Fig. 4. Reconstruction of Non-Convex Object by Three Methods. (a), (b), and (c) are three views of the original object. (d), (e), and (f) are reconstructed objects by the method in Figure 3, Figure 2 (c), and Figure 2 (a), respectively. (g), (h), and (i) are difference between the original object and reconstructed object. These results show that the line voting method for S^2 has advantages over established methods.

where N_8 is the eight-neighbourhood of the origin and $\oplus$ is the Minkowski addition of point sets on $\mathbf{R}^n$.

This theorem implies the next theorem.

Theorem 7. *If an object A is slice-convex with respect to axis $(1,0,0)^\top$, $(0,1,0)^\top$, and $(0,0,1)^\top$,*

$$\hat{A} \setminus A \subset (A \oplus N_{26} \oplus N_{26}) \setminus A. \tag{29}$$

From these theorems, in shape carving and visible voting, smoothing and weighting, respectively, are considered as operations to yield R' such that

$$|\hat{R}\Delta R'| > |R'\Delta R|, \; R \subseteq R \subset \hat{R}, \tag{30}$$

where

$$A\Delta B = (A \cap \overline{B}) \cup (\overline{A} \cap B) \tag{31}$$

and $|A|$ is the number of elements in set A.

In Figure 4, (a) is reconstructed objects by all lines which pass through detector voxels. (b) is reconstructed by a series of two-dimensional reconstruction. (c) is reconstructed by view cones whose apex moves on the plane $z = 0$. This configuration is widely used in computer vision.

Table 1 lists figures of $\hat{A}\Delta A$ and $\hat{A}\Delta A_2$ for three methods, where $A_2 = A \oplus N_{26} \oplus N_{26}$. Theorem 5 implies that for a convex object K, the relation $\Delta(\hat{K}, K_2) = 0$. This numerical result suggests that Theorem 5 would be valid for more large class of non-convex objects.

Table 1. Numbers of Different Voxels for $|A| = 70413$

Object	$\lvert\hat{A}\Delta A\rvert$	$\lvert\hat{A}\Delta A_2\rvert$
Fig. 4 (a)	5637	0
Fig. 4 (b)	6869	0
Fig. 4 (c)	12380	492

6 Conclusions

In this paper, we formulated shape from silhouettes in three- dimensional discrete space. This treatment of the problem implied an ambiguity theorem for the reconstruction of objects in discrete space.

References

1. Guggenheimer, H. W., *Applicable Geometry,* Robert E. Kniegen Pub. Inci, New York 1977.
2. Aloimonos, J., Visual shape computation, Proceedings of IEEE, **76**, (1988), 899-916.
3. Dobkin, D. P., Edelsbrunner, H., and Yap, C. K., Probing convex polytopes, Proc. 18th ACM Symposium on Theory of Computing, (1986), 424-432.
4. Campi, S., Reconstructing a convex surface from certain measurements of its projections, bollettio U.M.I., **6**, 945-959, 1986.
5. Boltyanski, V., Martin, H., and Soltan, P. S. *Excursions into Combinatorial Geometry,* Springer-Verlag; Berlin, 1997.
6. Kutulakos, K., Seitz,S. M., A theory of shape by space carving," Proceedings of 7th ICCV, **1**, 307-314, 1999.
7. Skiena,S. S., Interactive reconstruction via geometric probing, IEEE Proceedings, **80**, 1364-1383, 1992.

8. Skiena, S.S., Probing convex polygon with half-planes, Journal of Algorithms, **12**, 359-374, 1991.
9. Lindembaum, M., and Bruckstein, A., Reconstructing a convex polygon from binary perspective projections, Pattern Recognition, **23**, (1990), 1343-1350.
10. Laurentini, A., The visual hull concept for silhouette-bases image understanding, IEEE PAMI, **16**, (1994), 150-163.
11. Laurentini, A., How for 3D shape can be understood from 2D silhouettes, IEEE PAMI,**17**, (1995), 88-195.
12. Li, R. S.-Y. Reconstruction of polygons from projections, Information Processing Letters, **28**, 235-240, 1988.
13. Prince, J. L., Willsky, A.S., Reconstructing convex sets from support line measurements, IEEE Trans PAMI, **12**, 377-389, 1990.
14. Rao A. S., Goldberg, Y. K., Shape from diameter: Recognizing polygonal parts with parallel-jaw gripper, International Journal of Robotics Research, **13**, 16-37, 1994.
15. Kawamoto, K., Imiya, K., Detection of spatial points and lines by random sampling and voting process, Pattern Recognition Letters, **22**, 199-207, 2001.
16. Solmon, D. C., The X-ray transform, Journal of Math. Anal. and Appl. **56**, 61-83, 1976.
17. Hammaker, C., Smith, K. T., Solomon, D. C., Wagner, L., The divergent beam x-ray transform, Rocky Mountain Journal of Mathematics, **10**, 253-283, 1980.
18. Imiya, A., Kawamoto, K., Shape reconstruction from an image sequences, Lecture Notes in Computer Science, **2059**, 677-686, 2001.
19. Imiya, A., Kawamoto, K., Mathematical aspects of shape reconstruction from an image sequence, Proc. 1st Intl. Symp. 3D data Processing Visualization and Transformations, 632-635, 2002.

Characterization of the Closest Discrete Approximation of a Line in the 3-Dimensional Space

J.-L. Toutant

LIRMM - CNRS UMR 5506 - Universit de Montpellier II
161 rue Ada - 34392 Montpellier Cedex 5 - France
toutant@lirmm.fr

Abstract. The present paper deals with discrete lines in the 3-dimensional space. In particular, we focus on the minimal 0-connected set of closest integer points to a Euclidean line. We propose a definition which leads to geometric, arithmetic and algorithmic characterizations of naive discrete lines in the 3-dimensional space.

1 Introduction

In discrete geometry, as in Euclidean one, linear objects are essential. All the other discrete objects can be approximated as soon as the linear ones have been characterized. The only well understood class of linear discrete objects is the one of $(d-1)$-dimensional objects in the d-dimensional space, namely, the discrete hyperplanes [1,2]. Numerous works also exist on 1-dimensional linear discrete objects. They tackle this problem algorithmically [3,4] or arithmetically [5,6,7]. Moreover, discretization models, such as the standard [8] and the supercover [9] ones, define 2-connected discrete lines. Nevertheless, as far as we know, many problems are still open. For instance, we are currently unable to characterize the minimal 0-connected set of closest integer points to a Euclidean line in the 3-dimensional space.

In the present paper, our purpose is to introduce a modeling of discrete lines in the 3-dimensional space such that topological properties and the relationship with the closest integer points to the Euclidean line with same parameter are easily determined. We propose a representation of the 1-dimensional linear discrete object inspired by the notion of functionality [10,11]. Indeed, a connected discrete line in the 3-dimensional space should verify some conditions similar to this notion: we can define subsets of $\mathbb{Z}^3$ such that the discrete line is connected only if it contains at least a point of each of them.

The paper is organized as follows. In the next section, we recall some basic notions of discrete geometry useful to understand the remainder of the paper. Then, in the third section, we focus on already known naive discrete line. First we present the usual 2-dimensional definition which extends in higher dimensions to discrete hyperplanes. Secondly, the 3-dimensional current definition based on projections on particular planes is detailed. In the fourth section, we propose a definition related to the closest integer points to a Euclidean line and we introduce its geometric, arithmetic, and algorithmic characterizations in the 3-dimensional space.

2 Basic Notions

The aim of this section is to introduce the basic notions of discrete geometry used throughout the present paper. Let d be an integer greater than 1 and let $\{\mathbf{e_1}, \ldots, \mathbf{e_d}\}$ be the canonical basis of the Euclidean vector space $\mathbb{R}^d$. Let us call *discrete set* any subset of the *discrete space* $\mathbb{Z}^d$. The point $\mathbf{x} = \sum_{i=1}^{d} x_i \mathbf{e_i} \in \mathbb{R}^d$, with $x_i \in \mathbb{R}$ for each $i \in \{1, \ldots, d\}$, is represented by $(x_1, \ldots, x_d)$. A point $\mathbf{v} \in \mathbb{Z}^d$ is called a *voxel* in a d-dimensional space or a *pixel* in a 2-dimensional space.

Definition 1 (k-Adjacency or k-Neighborhood). *Let d be the dimension of the discrete space and $k \in \mathbb{N}$ such that $k < d$. Two voxels $\mathbf{v} = (v_1, \ldots, v_d)$ and $\mathbf{w} = (w_1, \ldots, w_d)$ are* k-neighbors *or* k-adjacent *if and only if:*

$$\|\mathbf{v}-\mathbf{w}\|_\infty = \max\{|v_1-w_1|, \ldots, |v_d-w_d|\} = 1 \text{ and } \|\mathbf{v}-\mathbf{w}\|_1 = \sum_{i=1}^{d} |v_i-w_i| \leq d-k.$$

Let $k \in \{0, \ldots, d-1\}$. A discrete set E is said to be *k-connected* if for each pair of voxels $(\mathbf{v}, \mathbf{w}) \in \mathrm{E}^2$, there exists a finite sequence of voxels $(\mathbf{s_1}, \ldots, \mathbf{s_p}) \in \mathrm{E}^p$ such that $\mathbf{v} = \mathbf{s_1}$, $\mathbf{w} = \mathbf{s_p}$ and the voxels $\mathbf{s_j}$ and $\mathbf{s_{j+1}}$ are k-neighbors, for each $j \in \{1, \ldots, p-1\}$.

Let E be a discrete set, $\mathbf{v} \in \mathrm{E}$ and $k \in \{0, \ldots, d-1\}$. The *k-connected component* of $\mathbf{v}$ in E is the maximal k-connected subset of E (with respect to set inclusion) containing $\mathbf{v}$.

Definition 2 (k-Separatingness). *A discrete set* E *is* k-separating *in a discrete set* F *if its complement in* F, $\overline{\mathrm{E}} = \mathrm{F} \setminus \mathrm{E}$, *has two distinct k-connected components.* E *is called a* separator *of* F.

Definition 3 (k-Simple Point, k-Minimality). *Let d be the dimension of the space and $k \in \mathbb{N}$ such that $k < d$. Let also* F *and* E *be two discrete sets such that* E *is k-separating in F. A voxel $\mathbf{v} \in \mathrm{E}$ is said to be* k-simple *if* $\mathrm{E} \setminus \{\mathbf{v}\}$ *remains k-separating in* F. *Moreover, a k-separating discrete set in* F *without k-simple points is said to be* k-minimal *in* F.

3 Discrete Lines

Lines are elementary objects in geometry. They have been widely studied in discrete geometry [1] and are the best known discrete objects. However we understand them in the 2-dimensional space as $(d-1)$-dimensional linear objects and not as 1-dimensional linear objects. Consequently, results on discrete lines in the 2-dimensional space extend in higher dimension to discrete hyperplanes and not to discrete lines in d-dimensional spaces. Definitions of discrete lines in the 3-dimensional space exist, but none are equivalent to the minimal 0-connected set of closest integer points to a Euclidean line.

3.1 The 2-Dimensional Space: Discrete Lines as Discrete Hyperplanes

First, discrete line drawing algorithms were designed to provide for the needs of digital plotters [12]. Later, arithmetic and geometric characterizations have been proposed [1,2]. The minimal 0-connected set of closest integer points to a Euclidean line is its closed naive representation [13]. The closed naive model introduced by E. Andres associates a Euclidean object $\mathcal{O}$ with the representation $\overline{\mathrm{N}}(\mathcal{O})$ defined as follows:

$$\overline{\mathrm{N}}(\mathcal{O}) = \left(\mathcal{B}^1\left(\frac{1}{2}\right) \oplus \mathcal{O}\right) \cap \mathbb{Z}^d, \tag{1}$$

$$= \left\{\mathbf{p} \in \mathbb{Z}^d; \left(\mathcal{B}^1\left(\frac{1}{2}\right) \oplus \mathbf{p}\right) \cap \mathcal{O} \neq \varnothing\right\}, \tag{2}$$

where $\mathcal{B}^1\left(\frac{1}{2}\right)$ is the ball of radius $\frac{1}{2}$ based on $\|\cdot\|_1$, and $\oplus$ denote the Minkowski sum:

$$A \oplus B = \{\mathbf{a} + \mathbf{b}; \mathbf{a} \in A \text{ and } \mathbf{b} \in B\}.$$

In Figure 1(a), an example of definition (1) is shown. The selected points are the ones contained in the band described by the translation of $\mathcal{B}_1\left(\frac{1}{2}\right)$ along the discrete line. In Figure 1(b), an example of definition (2) is shown. The selected discrete points are the ones for which the intersection between the ball $\mathcal{B}_1\left(\frac{1}{2}\right)$ centered on them and the Euclidean line is not empty. Both definitions are equivalent.

From an arithmetic point of view, the closed naive representation $\overline{\mathrm{N}}(\mathcal{D}(\mathbf{n}, \mu))$ of the Euclidean line $\mathcal{D}(\mathbf{n}, \mu)$ with normal vector $\mathbf{n} = (a, b) \in \mathbb{Z}^2$ and translation parameter μ is defined as follows:

$$\overline{\mathrm{N}}\left(\mathcal{D}(\mathbf{n})\right) = \left\{\mathbf{p} = (i, j) \in \mathbb{Z}^2; -\frac{\|\mathbf{n}\|_\infty}{2} \leq ai + bj + \mu \leq \frac{\|\mathbf{n}\|_\infty}{2}\right\} \tag{3}$$

Such an arithmetic representation is well adapted to the deduction of properties, such as the membership of a discrete point to a discrete line, and the definition of drawing and recognition algorithms.

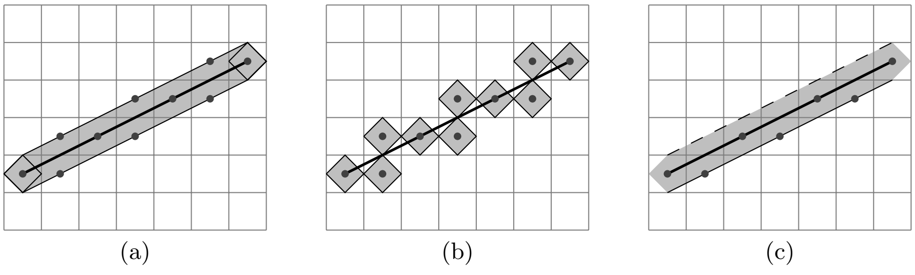


Fig. 1. (a) First definition of the closed naive representation of a line, (b) Second definition of the closed naive representation of a line, (c) The associated naive discrete line

However, the closed naive representation of an object can contain 0-simple points. A simple way to avoid such configuration is to restrict one of the inequalities in (3). By doing so, we obtain the naive line $\mathrm{N}\left(\mathcal{D}(\mathbf{n},\mu)\right)$ introduced by J.-P. Reveillès [1], defined as follows:

$$\mathrm{N}\left(\mathcal{D}(\mathbf{n},\mu)\right)=\left\{\mathbf{p}=(i,j)\in\mathbb{Z}^2; -\frac{\|\mathbf{n}\|_\infty}{2}\leq ai+bj+\mu<\frac{\|\mathbf{n}\|_\infty}{2}\right\} \quad (4)$$

Definition (4) is the common definition of discrete lines because it leads to the minimal 0-connected discrete set without simple points, as shown in Figure 1(c). J. Bresenham's line [12] is, in particular, a naive discrete line.

The above mentioned models of discrete lines easily extend in higher dimensions to discrete hyperplanes [1,2]. In particular, we have the following naive model of $\mathcal{P}(\mathbf{n},\mu)$, the hyperplane with normal vector $\mathbf{n}\in\mathbb{Z}^d$ and $\mu\in\mathbb{Z}$ its translation parameter:

$$\mathrm{N}\left(\mathcal{P}(\mathbf{n},\mu)\right)=\left\{\mathbf{v}=(v_1...,v_d)\in\mathbb{Z}^d; -\frac{\|\mathbf{n}\|_\infty}{2}\leq\sum_{i=1}^{d}n_iv_i+\mu<\frac{\|\mathbf{n}\|_\infty}{2}\right\} \quad (5)$$

In the 2-dimensional space, discrete lines are defined by their normal vector. This is not possible in higher dimensions: a line is then defined by its direction vector or its normal hyperplane. Another approach is necessary to understand them.

3.2 3-Dimensional Space: Discrete Lines and Projections

The closed naive description of a Euclidean line in the 3-dimensional space does not share all the properties of a Euclidean line in the 2-dimensional space. In particular, such a discrete set is not 0-connected.

Example 1. Let $\mathbf{v}=(1,1,2)$ be the direction vector of the Euclidean line $\mathcal{D}_{3D}(\mathbf{v})$ through the origin. Then, $\overline{\mathrm{N}}\left(\mathcal{D}_{3D}(\mathbf{v})\right)$, its closed naive representation is defined as follows:

$$\overline{\mathrm{N}}\left(\mathcal{D}_{3D}(\mathbf{v})\right)=\{p.(1,1,2);p\in\mathbb{Z}\}\,.$$

This set is obviously not connected.

Another definition have been proposed to characterize naive discrete lines in the 3-dimensional space. It was first introduced by A. Kaufman and E. Shimony [3] with an algorithm computing incrementally the set of its points. This algorithm is a generalization to the 3-dimensional space of the J. Bresenham's classical 2-dimensional one [12]. Considering that this naive line is provided with a direction vector $\mathbf{v}=(a,b,c)$ such that $\|\mathbf{v}\|_\infty=c\neq0$ and $\gcd(a,b,c)=1$, its projections on the planes normal to $\mathbf{e_1}$ and $\mathbf{e_2}$ (both equivalent to the 2-dimensional space $\mathbb{Z}^2$) are naive discrete lines. This simplification is the key point of the approach.

Later, I. Debled-Rennesson [7], O. Figueiredo and J.-P. Reveillès [5,6] proposed an arithmetic characterization of naive discrete lines in the 3-dimensional space.

$\mathrm{N}(\mathcal{D}_{3D}(\mathbf{v}))$, the naive representation of the Euclidean line $\mathcal{D}_{3D}(\mathbf{v})$ through the origin and directed by $\mathbf{v} = (a, b, c)$, such that $\|\mathbf{v}\|_\infty = c \neq 0$ and $\gcd(a, b, c) = 1$, is the set of discrete points $\mathbf{n} = (i, j, k) \in \mathbb{Z}^3$ verifying:

$$\begin{cases} -\frac{c}{2} \leq bk - cj < \frac{c}{2} \\ -\frac{c}{2} \leq ci - ak < \frac{c}{2} \end{cases} \tag{6}$$

This arithmetic definition characterizes the same set as A. Kaufman and E. Shimony's algorithm [6].

In [6], O. Figueiredo and J.-P. Reveillès notice that the resulting set is different from the one of the closest discrete points to the Euclidean line $\mathcal{D}_{3D}(\mathbf{v})$.

Example 2. Let $\mathbf{v} = (1, 2, 4)$ be the direction vector of the Euclidean line $\mathcal{D}_{3D}(\mathbf{v})$. Then, $\mathrm{N}(\mathcal{D}_{3D}(\mathbf{v}))$, the naive discrete representation of $\mathcal{D}_{3D}(\mathbf{v})$ is the set of points:

$$\mathrm{N}(\mathcal{D}_{3D}(\mathbf{v})) = \{p\mathbf{v} \oplus \{(0,0,0), (0,1,1), (1,1,2), (1,2,3)\}\,;\ p \in \mathbb{Z}\}\,.$$

The Euclidean distance between the points $\{p.\mathbf{v} \oplus (0,1,1);\ p \in \mathbb{Z}\}$ and the Euclidean line $\mathcal{D}_{3D}(\mathbf{v})$ is of 0.535, whereas the points $\{p\mathbf{v} \oplus (0,0,1);\ p \in \mathbb{Z}\}$ are only at a distance of 0.487 from the line. So the set of the closest points to the Euclidean line $\mathcal{D}_{3D}(\mathbf{v})$ is:

$$\{p.\mathbf{v} \oplus \{(0,0,0), (\mathbf{0},\mathbf{0},\mathbf{1}), (1,1,2), (1,2,3)\}\,;\ p \in \mathbb{Z}\}\,.$$

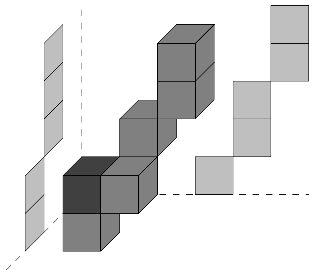


Fig. 2. The naive discrete line directed by $(1, 2, 4)$, for which projections are 2-dimensional naive discrete lines, contains an error in the point selection

Another definition of discrete lines was introduced by V. Brimkov and R. Barneva in [14]. Graceful lines are seen as intersection of particular discrete planes, the graceful ones. They are 0-connected but not minimal sets.

4 Discrete Lines as Sets of Closest Integer Points

The usual definition of discrete lines in 3-dimensional space is not satisfactory because it it not equivalent to the set of the closest integer points to the Euclidean line with same parameters and because geometric properties are lost. In the sequel, we propose a definition which overcomes these limitations and leads to geometric, arithmetic and algorithmic characterizations.

4.1 Minimal 0-Connected Set of Closest Integer Points

We are interested in the thinnest discrete line $\mathrm{D}_{3D}(\mathbf{v})$ through the origin and directed by $\mathbf{v} = (a, b, c) \in \mathbb{N}^3$ such that $\|\mathbf{v}\|_\infty = c \neq 0$ and $\gcd(a, b, c) = 1$. By thinnest, we mean the minimal (with respect to set inclusion) 0-connected set constituted by the closest discrete points to the Euclidean line $\mathcal{D}_{3D}(\mathbf{v})$.

Theorem 1. *The discrete line* $\mathrm{D}_{3D}(\mathbf{v})$ *is* 0*-connected if and only if :*

$$\forall k \in \mathbb{Z}, \exists (i, j) \in \mathbb{Z}^2;\ (i, j, k) \in \mathrm{D}_{3D}(\mathbf{v}).$$

Proof (Sketch). If $\exists k \in \mathbb{Z}$ such that $\nexists (i, j) \in \mathbb{Z}^2, (i, j, k) \in \mathrm{D}_{3D}(\mathbf{v})$ then $\mathrm{D}_{3D}(\mathbf{v})$ is not 0-connected since the discrete plane $\mathrm{P}(\mathbf{e_3}, k, 1)$ is 0-separating in $\mathbb{Z}^3$ and points of $\mathrm{D}_{3D}(\mathbf{v})$ belongs to both sides of it. Thus, if $\mathrm{D}_{3D}(\mathbf{v})$ is 0-connected, then $\forall k \in \mathbb{Z}, \exists (i, j) \in \mathbb{Z}^2, (i, j, k) \in \mathrm{D}_{3D}(\mathbf{v})$.

The discrete line $\mathrm{D}_{3D}(\mathbf{v})$ is the set of closest discrete points to the Euclidean line $\mathcal{D}_{3D}(\mathbf{v})$. Let us assume that $\mathbf{n}$ and $\mathbf{m} \in \mathbb{Z}^3$ belong to $\mathrm{D}_{3D}(\mathbf{v})$ such that $k_m = k_n + 1$. From the initial conditions on the direction vector $\mathbf{v}$ ($\|\mathbf{v}\|_\infty = c \neq 0$), the intersection between $\mathrm{D}_{3D}(\mathbf{v})$ and $\mathcal{P}(\mathbf{e_3}, k_m)$, the plane normal to $\mathbf{e_3}$ containing $\mathbf{m}$, is the intersection between $\mathrm{D}_{3D}(\mathbf{v})$ and $\mathcal{P}(\mathbf{e_3}, k_n)$ translated by vector $(\frac{a}{c}, \frac{b}{c}, 1)$. As the criterion is the distance to the line, in the worst case, $\mathbf{m} = \mathbf{n} + (1, 1, 1)$ and finally for each k_n, $\mathbf{n}$ and $\mathbf{m}$ are always 0-adjacent. Thus, if $\forall k \in \mathbb{Z}, \exists (i, j) \in \mathbb{Z}^2, (i, j, k) \in \mathrm{D}_{3D}(\mathbf{v})$, then $\mathrm{D}_{3D}(\mathbf{v})$ is 0-connected. $\square$

So for each $k \in \mathbb{Z}$, we look for the closest discrete points $\mathbf{n} = (i, j, k) \in \mathbb{Z}^3$ to $\mathrm{D}_{3D}(\mathbf{v})$. Let us now define $\mathcal{V}_{3D}$ (Figure 3(a)), a subset of $\mathbb{R}^3$ we use to determine them.

Definition 4. *Let* $\mathcal{P}(\mathbf{e_3}, 0)$ *be the Euclidean plane of normal vector* $\mathbf{e_3}$ *and with translation parameter* 0*. Then,* $\mathcal{V}_{3D}$ *is defined as follows:*

$$\mathcal{V}_{3D} = \left(\mathcal{B}^\infty\left(\frac{1}{2}\right) \setminus \left\{\mathbf{x} \in \mathbb{R}^3; \|\mathbf{x}\|_\infty = \frac{1}{2} \text{ and } sgn(x_1) = -sgn(x_2)\right\}\right) \cap \mathcal{P}(\mathbf{e_3}, 0).$$

Let $k \in \mathbb{Z}$. Let $\mathcal{P}(\mathbf{e_3}, k)$ be the Euclidean plane of normal vector $\mathbf{e_3}$ and with translation parameter k. Let $\mathcal{D}_{3D}(\mathbf{v})$ be the Euclidean line through the origin and directed by $\mathbf{v} = (a, b, c) \in \mathbb{N}^3$ such that $\|\mathbf{v}\|_\infty = c \neq 0$ and $\gcd(a, b, c) = 1$. Let $\mathbf{x} = \mathcal{D}_{3D}(\mathbf{v}) \cap \mathcal{P}(\mathbf{e_3}, k)$ be the intersection between the line $\mathcal{D}_{3D}(\mathbf{v})$ and the plane $\mathcal{P}(\mathbf{e_3}, k)$.

$\mathcal{V}_{3D}$ centered on $\mathbf{x}$ obviously contains at least one discrete point:

$$(\mathcal{V}_{3D} \oplus \mathbf{x}) \cap \mathbb{Z}^3 \neq \varnothing,$$

and:

Proposition 1. $\mathcal{V}_{3D}$ *centered on* $\mathbf{x}$ *contains the closest discrete points, included in* $\mathcal{P}(\mathbf{e_3}, k)$*, to* $\mathcal{D}_{3D}(\mathbf{v})$*:*

$$\forall \mathbf{n} \in (\mathcal{V}_{3D} \oplus \mathbf{x}) \cap \mathbb{Z}^3, d_2\left(\mathbf{n}, \mathcal{D}_{3D}(\mathbf{v})\right) = \min_{m \in \mathcal{P}(\mathbf{e_3}, k) \cap \mathbb{Z}^3} \left\{d_2\left(\mathbf{m}, \mathcal{D}_{3D}(\mathbf{v})\right)\right\} \quad (7)$$

Proof (Sketch). In order to prove this proposition, we have to evaluate the distance from the point $\mathbf{n}$ to the line $\mathcal{D}_{3D}(\mathbf{v})$. The cross product $\mathbf{v} \times \mathbf{n}$ is useful since its norm 2 is equal to this distance multiplied by $\|\mathbf{n}\|_2$. Then, points $\mathbf{x}$ are of the form $\left(\frac{ka}{c}, \frac{kb}{c}, k\right)$ and points $\mathbf{n}$ are of the form $\left(\frac{ka}{c} + \varepsilon_1, \frac{kb}{c} + \varepsilon_2, k\right)$. Those considerations are the key points to demonstrate the proposition. $\square$

This result allows to geometrically characterize the minimal 0-connected set $\mathrm{D}_{3D}(\mathbf{v})$ of the closest discrete points to the euclidean line $\mathcal{D}_{3D}(\mathbf{v})$.

Theorem 2. *The minimal* 0*-connected set* $\mathrm{D}_{3D}(\mathbf{v})$ *of the closest discrete points to the Euclidean line* $\mathcal{D}_{3D}(\mathbf{v})$ *with normal vector* $\mathbf{v} = (a, b, c) \in \mathbb{N}^3$ *such that* $\|\mathbf{v}\|_\infty = c \neq 0$ *and* $\gcd(a, b, c) = 1$ *is:*

$$\mathrm{D}_{3D}(\mathbf{v}) = (\mathcal{V}_{3D} \oplus \mathcal{D}_{3D}(\mathbf{v})) \cap \mathbb{Z}^3 \tag{8}$$
$$= \left\{\mathbf{p} \in \mathbb{Z}^3; (\mathcal{V}_{3D} \oplus \mathbf{p}) \cap \mathcal{D}_{3D}(\mathbf{v}) \neq \varnothing\right\} \tag{9}$$

Proof. Theorem 2 is a direct consequence of Proposition 1. $\mathcal{V}_{3D}$ is normal to $\mathbf{e_3}$. It can contains discrete points only if its component relative to $\mathbf{e_3}$ is an integer. In this particular case, $\mathcal{V}_{3D}$ contains only the discrete points of $\mathcal{P}(\mathbf{e_3}, k)$ for which the euclidean distance to the line $\mathcal{D}_{3D}(\mathbf{v})$ is minimal. Thus, for each $k \in \mathbb{Z}$, we select the points, at least one, closest to the line and obtain the minimal 0-connected set we are looking for. $\square$

4.2 Naive Discrete Lines in the 3-Dimensional Space

In order to obtain a naive discrete line in the 3-dimensional space, we arbitrarily select one discrete point when several are possible in $\mathrm{D}_{3D}(\mathbf{v})$. From the solution

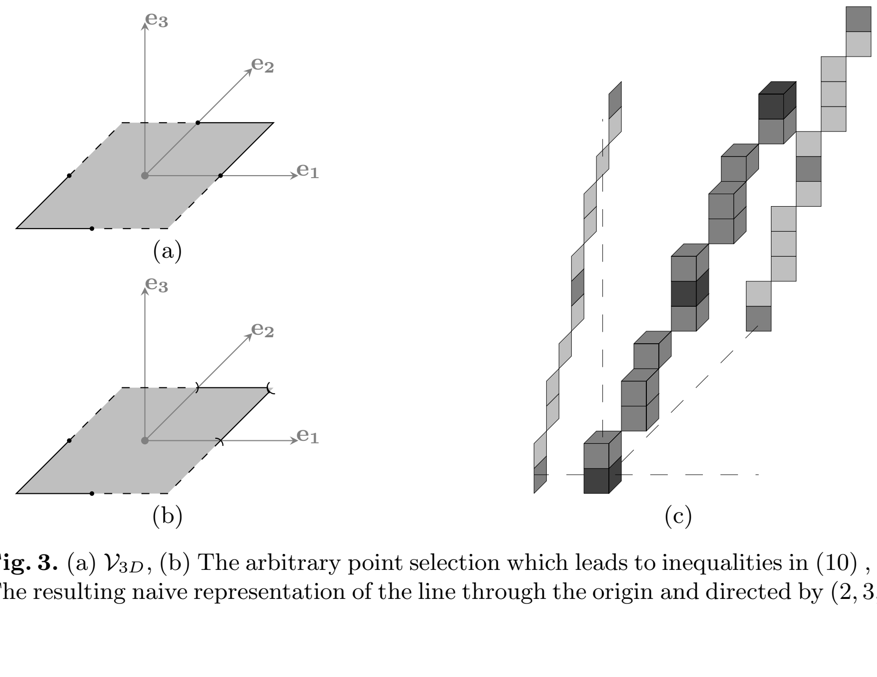


Fig. 3. (a) $\mathcal{V}_{3D}$, (b) The arbitrary point selection which leads to inequalities in (10) , (c) The resulting naive representation of the line through the origin and directed by $(2, 3, 6)$

shown in Figure 3(b) and the cross product $\mathbf{v} \times \mathbf{n}$ we deduce the following arithmetic definition. Only two components of the cross product are evaluated since the third one depends on them.

Definition 5. *Let* $\mathbf{v} = (a, b, c) \in \mathbb{N}^3$ *such that* $\|\mathbf{v}\|_\infty = c \neq 0$ *and* $\gcd(a, b, c) = 1$*. The naive discrete line through the origin and directed by* $\mathbf{v}$ *is the set of discrete points* $\mathbf{n} = (i, j, k) \in \mathbb{Z}^3$ *such that:*

$$\begin{cases} -\frac{c}{2} \leq & \operatorname{sgn}(ci - ak)\ (bk - cj) & < \frac{c}{2}, \\ -\frac{c}{2} \leq & \operatorname{sgn}(bk - cj)\ (ci - ak) & < \frac{c}{2}, \\ & (ci - ak, bk - cj) \neq \left(\frac{c}{2}, \frac{c}{2}\right) & \end{cases} \tag{10}$$

An example of the resulting set is shown in Figure 3(c) for the line directed by $(2, 3, 6)$. Its orthogonal projections on planes with normal vector $\mathbf{e_1}$ or $\mathbf{e_2}$ are not discrete lines.

Algorithm 1. Naive 3-dimensional discrete line drawing

Input : $\mathbf{v} = (a, b, c) \in \mathbb{N}^3$, $0 \leq a, b \leq c$, $c \neq 0$ and $gcd(a, b, c) = 1$.
Output : $\mathrm{D}_{3D}(\mathbf{v})$, the naive 3-dimensional discrete line trough the origin and directed by $\mathbf{v}$.

```
$i = 0,\ j = 0,\ k = 0;$
$p_1 = 0,\ p_2 = 0,\ p_3 = 0;$
select$(i, j, k)$;
**for** $k = 1$ to $c$ **do**
   $p_1 = p_1 + b;$
   $p_2 = p_2 - a;$
   **if** $|p_1| > |p_1 - c|$ **then**
      $p_1 = p_1 - c;$
      $p_3 = p_3 + a;$
      $j++;$
   **end if**
   **if** $|p_2| > |p_2 + c|$ **then**
      $p_2 = p_2 + c;$
      $p_3 = p_3 - b;$
      $i++;$
   **end if**
   **if** $|p_1| = |p_1 - c|$ and $|p_3| > |p_3 + a|$ **then**
      $p_1 = p_1 - c;$
      $p_3 = p_3 + a;$
      $j++;$
   **end if**
   **if** $|p_2| = |p_2 + c|$ and $|p_3| > |p_3 - b|$ **then**
      $p_2 = p_2 + c;$
      $p_3 = p_3 - b;$
      $i++;$
   **end if**
   select$(i, j, k)$;
**end for**
```

From the arithmetic definition (10), we design a simple drawing algorithm described in Algorithm 1. We use the three components (p_1, p_2, p_3) of the cross product $\mathbf{v} \times \mathbf{n}$ to evaluate the distance from $\mathbf{n} = (i, j, k)$ to the line, and not only p_1 and p_2 as in the arithmetic definition. p_3 allows us to determine incrementally which of the inequality should be considered large without studying the sign of p_1 and p_2. First, the algorithm is initiated with a trivial point of the discrete set, $(0, 0, 0)$ for which $\mathbf{v} \times \mathbf{n} = \mathbf{0}$. Then k will be incremented until it reaches the value c. At each step, four conditions are successively evaluated. Both first are true if the bounds of inequalities in (10) are not concerned. They check if the current point (i, j, k) is close to the line or if it has to be updated by incrementing either i or j or the both. The two last conditions concern the bound of the inequalities in (10). To choose between them without studying the sign of p_1 or p_2, we just study the distance to the line. When $|p_1| = |p_1 - c|$, changing p_1 has no influence on the distance to the line. Moreover, p_2 is fixed and so do not change the distance either. So, in order to minimize the distance, it is then sufficient to minimize p_3. That's what is done with the two last conditions. The exclusion of the point $\mathbf{n} = (i, j, k) \in \mathbb{Z}^3$ such that $(bk - cj, ci - ak) = (\frac{c}{2}, \frac{c}{2})$ is a consequence of the strict inequalities $|p_3| > |p_3 + a|$ and $|p_3| > |p_3 - b|$.

5 Conclusion

In the present paper, we have proposed a definition of naive discrete lines in the 3-dimensional space and given geometric, arithmetic and algorithmic characterizations. The resulting set is the minimal 0-connected set of the closest integer points to a Euclidean line. This is a significant property since we expect from a discretization that it approximates as close as possible its Euclidean equivalent. Previous definitions are unable to fulfill this requirement. Indeed, in order to simplify the original 3-dimensional problem, they reduce it to the determination of the discrete points belonging to two naive lines in the 2-dimensional space and thus loose relationship between the different directions of the space. Our definition provide naive 3-dimensional discrete line with new geometric properties. We recover the intrinsic symmetry of line in case where $\mathrm{D}_{3D}(\mathbf{v})$ does not contain simple points. At the opposite, the projections on the planes normal to the vectors of the basis do not correspond to any particular discrete sets. Consequently, the representation of discrete lines as intersections of discrete planes do not seem compatible with our approach.

The study of 3-dimensional discrete line not only concern naive ones. The determination of the best k-connected approximation of a Euclidean line is also of interest. In the same way, trying to extend results to the d-dimensional case could confirm or invalidate our approach.

The closed naive model allows discretizations of hyperplanes with geometric and topological properties. It seems that it is also the case for the largest class of $(d-1)$-dimensional objects as hyperspheres. At the opposite, it leads to nothing when apply on objects of other dimensions. The appropriated discretization model certainly depends on the dimension of the considered object. It would be interesting to applied our discretization scheme to other planar objects like circles.

References

1. Reveillès, J.P.: Géométrie discrète, calcul en nombres entiers et algorithmique. Thèse d'Etat, Université Louis Pasteur, Strasbourg (1991)
2. Andres, E., Acharya, R., Sibata, C.: Discrete analytical hyperplanes. CVGIP: Graphical Models and Image Processing **59** (1997) 302–309
3. Kaufman, A., Shimony, E.: 3d scan-conversion algorithms for voxel-based graphics. In: SI3D '86: Proceedings of the 1986 workshop on Interactive 3D graphics, New York, NY, USA, ACM Press (1987) 45–75
4. Cohen-Or, D., Kaufman, A.: 3d line voxelization and connectivity control. IEEE Comput. Graph. Appl. **17** (1997) 80–87
5. Figueiredo, O., Reveilles, J.: A contribution to 3d digital lines (1995)
6. Figueiredo, O., Reveillès, J.P.: New results about 3D digital lines. In Melter, R.A., Wu, A.Y., Latecki, L., eds.: Vision Geometry V. Volume 2826. (1996) 98–108
7. Debled-Rennesson, I.: Etude et reconnaissance des droites et plans discrets. Thèse de Doctorat, Universit Louis Pasteur, Strasbourg. (1995)
8. Andres, E.: Discrete linear objects in dimension n: the standard model. Graphical Models **65** (2003) 92–111
9. Brimkov, V., Andres, E., Barneva, R.: Object discretizations in higher dimensions. Pattern Recognition Letters **23** (2002) 623–636
10. Debled-Rennesson, I., Reveillès, J.P.: A new approach to digital planes. In: Vision Geometry III, Proc. SPIE. Volume 2356., Boston, USA (1994)
11. Berthe, V., Fiorio, C., Jamet, D.: Generalized functionality for arithmetic discrete planes. In: DGCI'05. (2005)
12. Bresenham, J.: Algorithm for computer control of a digital plotter. IBM Systems Journal **4** (1965) 25–30
13. Andres, E.: Modélisation Analytique Discrète d'Objets Géométriques. Habilitation à diriger des recherches, UFR Sciences Fondamentale et Appliquées - Université de Poitiers (France) (2000)
14. Brimkov, V., Barneva, R.: Graceful planes and lines. Theoritical Computer Science **283** (2002) 151–170

Margin Maximizing Discriminant Analysis for Multi-shot Based Object Recognition

Hui Kong[1], Eam Khwang Teoh[2], and Pengfei Xu[1]

[1] Panasonic Singapore Labs
Blk 1022 Tai Seng Ave., Singapore 534415
{hui.kong, pengfei.xu}@sg.panasonic.com
[2] School of Electrical and Electronic Engineering
Nanyang Technological University, Singapore 639798
eekteoh@ntu.edu.sg

Abstract. This paper discusses general object recognition by using image set in the scenario where multiple shots are available for each object. As a way of matching sets of images, canonical correlations offer many benefits in accuracy, efficiency, and robustness compared to the classical parametric distribution-based and non-parametric sample-based methods. However, it is essentially an representative but not a discriminative way for all the previous methods in using canonical correlations for comparing sets of images. Our purpose is to define a transformation such that, in the transformed space, the sum of canonical correlations (the cosine value of the principle angles between any two subspaces) of the intra-class image sets can be minimized and meantime the sum of canonical correlations of the inter-class image sets can be maximized. This is done by learning a margin-maximized linear discriminant function of the canonical correlations. Finally, this transformation is derived by a novel iterative optimization process. In this way, a discriminative way of using canonical correlations is presented. The proposed method significantly outperforms the state-of-the-art methods for two different object recognition problems on two large databases: a celebrity face database which is constructed using Image Google and the ALOI database of generic objects where hundreds of sets of images are taken at different views.

1 Introduction

In this paper, we consider multiple-shot based object recognition in the general settings where a set or several sets of images of an object is available. See Figure 1 for examples of two sets of face patterns of the same subject. The objective of this work is to efficiently classify a novel set of images to one of the training classes, each represented by one or several sets of vectors. This work does not explicitly investigate any temporal information between consecutive images or semantics in images as in [14, 15], but is solely based on learning the labeled image sets. Therefore, we expect that the proposed method can be applied to many other problems requiring a set comparison.

The previous related approaches for set matching can be broadly partitioned into parametric-distribution model-based and pair-wise sample-based methods. In the model-based approaches [1, 2], each set is represented by a parametric distribution model, typically Gaussian. The closeness of the two distributions is then measured by the Kullback-Leibler Divergence (KLD) distance. However, these methods easily fail due to the difficulty of parameter estimation with limited training data. The other way is based on the matching of pair-wise samples of sets, e.g. Nearest Neighbor (NN) or Hausdorff-distance matching [7]. However, they often suffer from outliers (noise) as well as the natural variability of the sensory data due to the 3D nature of the observed objects. In addition, such methods are extremely computationally expensive.

Recently, canonical angles [12, 13] has attracted increasing attention for image-set matching [9-11]. Each set is represented by a linear subspace and the angles between two subspaces are exploited as a similarity measure of two sets. Compared with distribution- and sample-based matching methods, the benefits of canonical angles have been noted in [2,11]. The kernelized canonical angles was proposed in [10]. A heuristic Constrained Mutual Subspace Method (CMSM) was proposed in [11]. In CMSM, a constrained subspace is defined as the subspace in which the entire class population exhibits small variance. It was showed that the sets of different classes in the constrained subspace had large canonical angles. However, the principle of CMSM is rather heuristic, especially the process of selecting the dimensionality of the constrained subspace.

Although the previous canonical-angle based methods have achieved some promising results, however, it is essentially not a discriminative but only a representative way of all the previous methods to use canonical angles for comparing sets of images. This paper presents a novel margin maximizing linear discriminant analysis of image sets based on canonical angles, called Margin Maximizing Discriminant Analysis of Canonical Correlations (MMCC). Our goal is to derive a transformation such that, in the transformed space, the sum of canonical correlations of the intra-class image sets can be minimized and meantime the sum of canonical correlations of the inter-class image sets can be maximized. This is done by learning a margin-maximized linear discriminant function of the canonical correlations. The linear mapping is solved using a novel iterative optimization algorithm. The discriminative capability of the proposed method is shown to be significantly better than the method [7] that simply aggregates canonical correlations and the k-NN methods in the LDA (linear discriminant analysis) subspace [7]. Compared with CMSM [11], the proposed method achieve higher accuracy and is more practical due to its simplicity in feature selection. In addition, it is more theoretically appealing.

2 Canonical Correlations

Canonical correlations, the cosines of canonical angles $0 \leq \theta_1 \leq \cdots \leq \theta_d \leq \frac{\pi}{2}$ between any two d-dimensional linear subspaces, $\mathcal{L}_1$ and $\mathcal{L}_2$, are defined as:

$$\cos\theta_i = \max_{\mathbf{u}_i \in \mathcal{L}_1} \max_{\mathbf{v}_i \in \mathcal{L}_2} \mathbf{u}_i^T \mathbf{v}_i \tag{1}$$

subject to $\mathbf{u}_i^T\mathbf{u}_i = \mathbf{v}_i^T\mathbf{v}_i = 1$, $\mathbf{u}_i^T\mathbf{u}_j = \mathbf{v}_i^T\mathbf{v}_j = 0$, for $i \neq j$. The SVD solution [14] is more numerically stable compared with the others, and it is as follows: Assume that $\mathbf{P}_1 \in \mathbf{R}^{n\times d}$ and $\mathbf{P}_2 \in \mathbf{R}^{n\times d}$ form unitary orthogonal basis matrices for two linear subspaces, $\mathcal{L}_1$ and $\mathcal{L}_2$. Let the SVD of $\mathbf{P}_1^T\mathbf{P}_2$ be

$$\mathbf{P}_1^T\mathbf{P}_2 = \mathbf{Q}_{12}\Lambda\mathbf{Q}_{21}^T \quad s.t. \quad \Lambda = diag(\sigma_1, \cdots, \sigma_d) \tag{2}$$

where $\mathbf{Q}_{12}$ and $\mathbf{Q}_{21}$ are orthogonal matrices, i.e. $\mathbf{Q}_{ij}^T\mathbf{Q}_{ij} = \mathbf{Q}_{ij}\mathbf{Q}_{ij}^T = \mathbf{I}_d$. Canonical correlations are $\{\sigma_1, \cdots, \sigma_d\}$ and the associated canonical vectors are $\mathbf{U} = \mathbf{P}_1\mathbf{Q}_{12} = [\mathbf{u}_1, \ldots, \mathbf{u}_d]$, $\mathbf{V} = \mathbf{P}_2\mathbf{Q}_{21} = [\mathbf{v}_1, \ldots, \mathbf{v}_d]$. The canonical correlations tell us how close are the closest vectors of two subspaces. See Figure 2 for the canonical vectors computed from the sample image sets given in Figure 1. Although the principal components vary for different imaging conditions of the sets, the canonical vectors well capture the common modes of the two different sets.

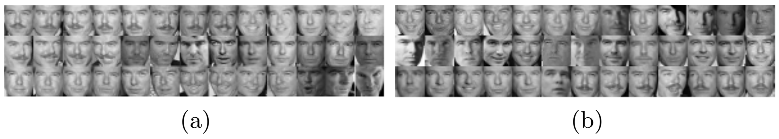


Fig. 1. Two sets of face images of Pierce Brosnan

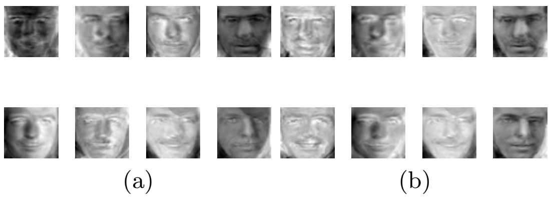


Fig. 2. Principal components vs. canonical vectors. (a) The first 4 principal components computed from the two image sets shown in Figure 1. The principal components of the different image sets (see each column) show significantly different variations even for the same objects. (b) The first 4 canonical vectors of the two image sets. Every pair of canonical vectors (each column) well captures the common modes (views and illuminations) of the two sets, i.e. the pairwise canonical vectors are almost similar.

3 Margin Maximizing Discriminant Analysis of Canonical Correlations

Assume m sets of vectors are given as $\{\mathbf{X}_1, \ldots, \mathbf{X}_m\}$, where $\mathbf{X}_i$ describes the i-th set containing n-dimensional observation vectors (or images) in its columns. Each set belongs to one of object classes, $\mathcal{C}_i$. A d-dimensional linear subspace of the i-th set is represented by an orthonormal basis matrix $\mathbf{P}_i \in \mathcal{R}^{n\times d}$ s.t. $\mathbf{X}_i\mathbf{X}_i^T \simeq \mathbf{P}_i\Lambda_i\mathbf{P}_i^T$, where Λ_i, $\mathbf{P}_i$ are the eigenvalue and eigenvector matrices of the d largest eigenvalues respectively. We define a transformation matrix $\mathbf{T}$, $\mathbf{T}$:

$\mathbf{X}_i \longrightarrow \mathbf{Y}_i = \mathbf{T}^T\mathbf{X}_i$, so that the transformed image sets are more class-wise discriminative using canonical correlations.

The orthonormal basis matrices of the subspaces for the transformed data are obtained from the previous matrix factorization of $\mathbf{X}_i\mathbf{X}_i^T$:

$$\mathbf{Y}_i\mathbf{Y}_i^T = (\mathbf{T}^T\mathbf{X}_i)(\mathbf{T}^T\mathbf{X}_i)^T \simeq (\mathbf{T}^T\mathbf{P}_i)\Lambda_i(\mathbf{T}^T\mathbf{P}_i)^T \tag{3}$$

Generally, $\mathbf{T}^T\mathbf{P}_i$ is not an orthonormal basis matrix. Note that canonical correlations are defined only for orthonormal basis matrices of subspaces. Therefore, a normalization process of $\mathbf{P}_i$ to $\mathbf{P}_i^{'}$ is performed as follows so that $\mathbf{T}^T\mathbf{P}_i^{'}$ can represent an orthonormal basis matrix of the transformed data. For normalization, QR-decomposition of $\mathbf{T}^T\mathbf{P}_i$ is first performed s.t. $\mathbf{T}^T\mathbf{P}_i = \Phi_i\Delta_i$, where $\Phi_i \in \mathcal{R}^{n\times d}$ is the orthonormal matrix with the first d columns and $\Delta_i \in \mathcal{R}^{d\times d}$ is the invertible upper-triangular matrix with the first d rows. From (3), $\mathbf{Y}_i = \mathbf{T}^T\mathbf{P}_i\sqrt{\Lambda_i} = \Phi_i\Delta_i\sqrt{\Lambda_i}$. As $\Delta_i\sqrt{\Lambda_i}$ is still an upper-triangular matrix, Φ_i can represent an orthonormal basis matrix of the transformed data $\mathbf{Y}_i$. As Δ_i is invertible,

$$\Phi_i = \mathbf{T}^T(\mathbf{P}_i\Delta_i^{-1}) \longrightarrow \mathbf{P}_i^{'} = \mathbf{P}_i\Delta_i^{-1} \tag{4}$$

The similarity of any two transformed data sets are defined as the sum of canonical correlations by

$$\mathcal{F}_{ij} = \max_{\mathbf{Q}_{ij},\mathbf{Q}_{ji}} tr(\mathbf{M}_{ij}) \tag{5}$$

$$\mathbf{M}_{ij} = \mathbf{Q}_{ij}^T\mathbf{P}_i^{'T}\mathbf{T}\mathbf{T}^T\mathbf{P}_j^{'}\mathbf{Q}_{ji} \quad or \quad \mathbf{M}_{ij} = \mathbf{T}^T\mathbf{P}_j^{'}\mathbf{Q}_{ji}\mathbf{Q}_{ij}^T\mathbf{P}_i^{'T}\mathbf{T} \tag{6}$$

as $tr(\mathbf{AB}) = tr(\mathbf{BA})$ for any matrix $\mathbf{A}$ and $\mathbf{B}$. $\mathbf{Q}_{ij}$ and $\mathbf{Q}_{ji}$ are the rotation matrices defined in the solution of canonical correlations (2).

Maximum Margin Criterion (MMC) [19] is a recently proposed feature extraction criterion. This new criterion is general in the sense that when combined with a suitable constraint it can actually give rise to the most popular feature extractor in the literature, i.e. Linear Discriminant Analysis. Using the same representation as LDA, the goal of MMC is to maximize the criterion $\mathcal{J}(\mathbf{W}) = \mathbf{W}^T(\mathbf{S}_b - \mathbf{S}_w)\mathbf{W}$. Although both MMC and LDA [20] are supervised subspace learning approaches, the new feature extractors derived from MMC do not suffer from the Small Sample Size problem [20], which is known to cause serious stability problems for LDA (based on Fisher's criterion). Compared with the other LDA-based methods, e.g., Fisherface [3], Null-space based LDA [4,5], Direct-LDA [6], MMC-based discriminant analysis does not discard those useful discriminative information embedded in some specific subspaces[1] to ensure the stable solutions. In addition, the computation of MMC is easier than that of LDA since MMC does not have inverse operation. The projection matrix $\mathbf{W}$ can be obtained by solving the following eigenvector decomposition problem:

$$(\mathbf{S}_b - \mathbf{S}_w)\mathbf{W} = \mathbf{W}\Lambda \tag{7}$$

[1] These discarded subspaces are: the null-space of $\mathbf{S}_w$ in [3], the range space of $\mathbf{S}_w$ in [4,5] and the null-space of $\mathbf{S}_b$ in [6].

The transformation $\mathbf{T}$ is found to maximize the similarities of any pairs of sets of inter-classes while minimizing the similarities of pairwise sets of intra-classes. Matrix $\mathbf{T}$ is defined by

$$\mathbf{T} = \arg\max_{\mathbf{T}} (\sum_{i=1}^{m} \sum_{l \in \mathcal{B}_i} \mathcal{F}_{il} - \sum_{i=1}^{m} \sum_{k \in \mathcal{W}_i} \mathcal{F}_{ik}) \tag{8}$$

where $\mathcal{B}_i = \{j | \mathbf{X}_j \in \bar{\mathcal{C}}_i\}$ and $\mathcal{W}_i = \{j | \mathbf{X}_j \in \mathcal{C}_i\}$. That is, the two sets $\mathcal{W}_i$ and $\mathcal{B}_i$ denote the within-class and between-class sets of a given set class i respectively, which are similarly defined with [3].

The optimization problem of $\mathbf{T}$ involves the variables of $\mathbf{Q}$, $\mathbf{P}'$ as well as $\mathbf{T}$. As the other variables are not explicitly represented by $\mathbf{T}$, a closed form solution for $\mathbf{T}$ is hard to find. We propose an iterative optimization algorithm. Thus, the proposed iterative optimization is comprised of the three main steps: normalization of $\mathbf{P}$, optimization of matrices $\mathbf{Q}$ and $\mathbf{T}$. The normalization of $\mathbf{P}$ has been introduced above, and the optimization of $\mathbf{Q}$ and $\mathbf{T}$ is as follows:

Table 1. Margin Maximized Discriminant Analysis of Canonical Correlations

```
Algorithm: Margin Maximized Discriminant Analysis of Canonical Correlations
1. Input: All $\mathbf{P}_i \in \mathcal{R}^{n \times d}$
2. $\mathbf{T} \longleftarrow \mathbf{I}_n$
3. Do the iterations as follows:
4. For each $i$, do QR-decomposition: $\mathbf{T}^T\mathbf{P}_i = \Phi_i \Delta_i \longrightarrow \mathbf{P}_i' = \mathbf{P}_i \Delta_i^{-1}$
5. For each pair, $i, j$, do SVD: $\mathbf{P}_i'^T \mathbf{T}\mathbf{T}^T \mathbf{P}_i' = \mathbf{Q}_{ij} \Lambda \mathbf{Q}_{ji}^T$
6. Compute $\mathbf{S}_b' = \sum_{i=1}^{m} \sum_{k \in \mathcal{B}_i} (\mathbf{P}_l'\mathbf{Q}_{li} - \mathbf{P}_i'\mathbf{Q}_{il})(\mathbf{P}_l'\mathbf{Q}_{li} - \mathbf{P}_i'\mathbf{Q}_{il})^T$,
           $\mathbf{S}_w' = \sum_{i=1}^{m} \sum_{k \in \mathcal{W}_i} (\mathbf{P}_k'\mathbf{Q}_{ki} - \mathbf{P}_i'\mathbf{Q}_{ik})(\mathbf{P}_k'\mathbf{Q}_{ki} - \mathbf{P}_i'\mathbf{Q}_{ik})^T$
7. Solve the eigenvalue problem of $(\mathbf{S}_b' - \mathbf{S}_w')\mathbf{T} = \mathbf{T}\Lambda$.
8. Output: $\mathbf{T} \in \mathcal{R}^{n \times n}$
```

Rotation matrices $\mathbf{Q}_{ij}$ for every i, j are obtained for a fixed $\mathbf{T}$ and $\mathbf{P}_i'$. The correlation matrix $\mathbf{M}_{ij}$ in the left of (5) can be conveniently used for the optimization of $\mathbf{Q}_{ij}$, as it has $\mathbf{Q}_{ij}$ outside of the matrix product. Let the SVD of $\mathbf{P}_i'^T \mathbf{T}\mathbf{T}^T \mathbf{P}_i'$ be

$$\mathbf{P}_i'^T \mathbf{T}\mathbf{T}^T \mathbf{P}_i' = \mathbf{Q}_{ij} \Lambda \mathbf{Q}_{ji}^T \tag{9}$$

where Λ is a diagonal matrix, and $\mathbf{Q}_{ij}$ and $\mathbf{Q}_{ji}$ are orthogonal rotation matrices.

The optimal discriminant transformation $\mathbf{T}$ is computed for given $\mathbf{P}_i'$ and $\mathbf{Q}_{ij}$ by using the definition of $\mathbf{M}_{ij}$ in the right of (5) and (6). With $\mathbf{T}$ being on the outside of the matrix product, it is convenient to solve for. The discriminative function is found by

$$\mathbf{T} = \arg\max_{\mathbf{T}} tr(\mathbf{T}^T(\mathbf{S}_b - \mathbf{S}_w)\mathbf{T}) \tag{10}$$

$$\mathbf{S}_b = \sum_{i=1}^{m} \sum_{l \in \mathcal{B}_i} \mathbf{P}_l' \mathbf{Q}_{li} \mathbf{Q}_{il}^T \mathbf{P}_i'^T, \quad \mathbf{S}_w = \sum_{i=1}^{m} \sum_{k \in \mathcal{W}_i} \mathbf{P}_k' \mathbf{Q}_{ki} \mathbf{Q}_{ik}^T \mathbf{P}_i'^T \tag{11}$$

where $\mathcal{W}_i = \{j|\mathbf{X}_j \in \mathcal{C}_i\}$ and $\mathcal{B}_i = \{j|\mathbf{X}_j \in \bar{\mathcal{C}}_i\}$. For a more stable solution, an alternative optimization is finally proposed by

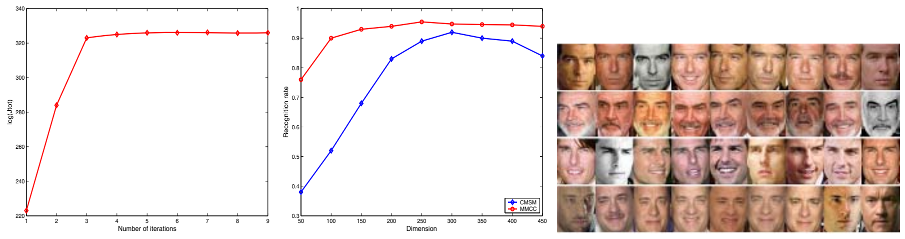
$$\mathbf{T} = \arg\max_{\mathbf{T}} tr(\mathbf{T}^T(\mathbf{S}_b^{'} - \mathbf{S}_w^{'})\mathbf{T}) \tag{12}$$

where $\mathbf{S}_b^{'} = \sum_{i=1}^{m} \sum_{k\in\mathcal{B}_i} (\mathbf{P}_l^{'}\mathbf{Q}_{li} - \mathbf{P}_i^{'}\mathbf{Q}_{il})(\mathbf{P}_l^{'}\mathbf{Q}_{li} - \mathbf{P}_i^{'}\mathbf{Q}_{il})^T$ and $\mathbf{S}_w^{'} = \sum_{i=1}^{m} \sum_{k\in\mathcal{W}_i}$ $(\mathbf{P}_k^{'}\mathbf{Q}_{ki} - \mathbf{P}_i^{'}\mathbf{Q}_{ik})(\mathbf{P}_k^{'}\mathbf{Q}_{ki} - \mathbf{P}_i^{'}\mathbf{Q}_{ik})^T$.

Note that no loss of generality is incurred by this modification of the objective function as $\mathbf{A}^T\mathbf{B} = \mathbf{I} - \frac{1}{2}(\mathbf{A}-\mathbf{B})^T(\mathbf{A}-\mathbf{B})$ if $\mathbf{A} = \mathbf{T}^T\mathbf{P}_i^{'}\mathbf{Q}_{ij}$ and $\mathbf{A} = \mathbf{T}^T\mathbf{P}_j^{'}\mathbf{Q}_{ji}$. The solution of Eq.10 is obtained by solving the following eigenvalue problem,

$$(\mathbf{S}_b^{'} - \mathbf{S}_w^{'})\mathbf{T} = \mathbf{T}\Lambda \tag{13}$$

Note that the proposed margin maximizing discriminant learning criterion can avoid a singular case of $\mathbf{S}_w^{'}$ as in Fisher Linear Discriminant (FLD) [3] where the Fisher criterion is adopted. No additional steps are required to accommodate the singularity problem as in [3,4,5,6] and, therefore, no discriminant information is lost. In addition, the computation complexity becomes simpler by avoiding solving the inverse of $\mathbf{S}_w^{'}$. With the identity matrix $\mathbf{I} \in \mathcal{R}^{n\times n}$ as the initial value



(a) (b) (c)

Fig. 3. (a) Convergence after several iterations. (b) Dimensionality selection for the proposed method and CMSM. The proposed method is more favorable than CMSM in dimensionality selection. CMSM shows a high peaking. (c) Sample images from the **Celebrity Face Database**.

of $\mathbf{T}$, the algorithm is iterated until it converges to a stable point. A Pseudo-code for the learning is given in Table 1. See Figure 3 (a), it converges fast and stably. After $\mathbf{T}$ is found to maximize the sum of canonical correlations of inter-class sets and minimize that of intra-class sets, a comparisons of any two sets is achieved using the similarity value defined in Eq.4.

4 Experimental Results and Discussion

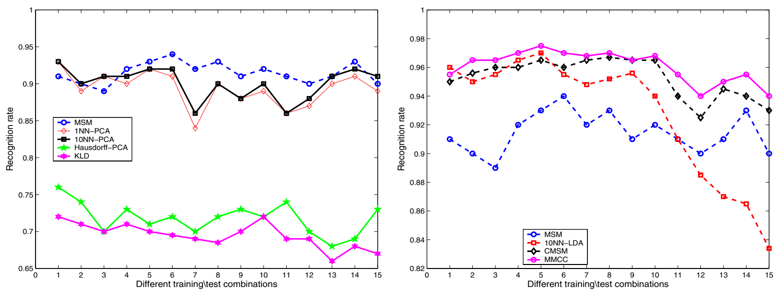
4.1 Experimental Setting for Face Recognition

We have acquired a database called **Celebrity Face Database** for 160 celebrities all over the world, see Fig.3 (c) for some sample images. All of these face images are obtained using Google image searching engine. 240 face images are gathered for each person with significant variations in pose, illumination and expressions. All face images are rotated roughly and normalized to the size of 20×20 according to the eye positions. We divide the 240 face images of the same person into 6 groups with each group containing 40 face images (some groups have larger variations in lighting conditions than others). For each person, two groups of face images are used as training sets for the construction of within-class sets and between-class sets, and the left four groups are for testing. Therefore, altogether 15 ($\mathcal{C}_6^2$) training/test combinations are formed to examine the performance of recognition. We compared the performance of MMCC algorithm to that of the following algorithms: (1) K-L Divergence algorithm (KLD) [1]; (2) k-Nearest Neighbours (k-NN) and Hausdorff distance in (i) PCA, and (ii) LDA [3] subspaces estimated from training data [7]; (3) Mutual Subspace Method (MSM) [9], which is equivalent to the simple aggregation of canonical correlations; (4) Constrained MSM (CMSM) [11] used in a commercial system FacePass [21].

In KLD, 95% of data energy was explained by the principal subspace of training data used. In NN-PCA, the optimal number of principal components was 150 without the first three. In NN-LDA, PCA with 150 dimensions (removal of the first 3 principal components did not improve the LDA performance) was applied first to avoid singularity problems and the best dimension of LDA subspace was 150 again. In both MSM and CMSM, the PCA dimension of each image set was fixed to 10, which represents more than 98% of data energy of the set. All 10 canonical correlations were exploited. In CMSM, the best dimension of the constrained subspace was found to be 300 in terms of the test recognition rates as shown in Figure 3 (b). The CMSM exhibits a peaking and does not have a principled way of choosing dimensionality of the constrained subspace in practice. By contrast, the proposed method provided constant recognition rates regardless of dimensionality of $\mathbf{T}$ beyond a certain point, as shown in Figure 3 (b). Thus we could fix the dimensionality at 250 for all experiments. This behavior is highly beneficial from the practical point of view. The PCA dimension of image sets was also fixed to 10 for the proposed method.

The 15 experiments were arranged in the order of increasing K-L Divergence between the training and test data. Lower K-L Divergence indicates more similar conditions. The recognition rates of the evaluated algorithms is shown in Figure 4. First, different methods of measuring set similarity were compared in Figure 4 (a). Most of the methods generally had lower recognition rates for experiments having larger KL-Divergence. The KLD method achieved by far the worst recognition rate. Considering the arbitrary lighting and viewing conditions across data, the distribution of within-class face patterns was very broad, making this result unsurprising. As representatives of non-parametric sample-based matching, the 1-NN, 10-NN, and Hausdorff distance methods defined in the PCA subspace were evaluated. It was observed that the Hausdorff-distance measure provided consistent but far poorer results than the NN methods. 10-NN yielded the best accuracy of the three, which is worse than MSM by 7.5% on average.

Its performance greatly varied across the experiments while MSM showed robust performance under the different experimental conditions.



(a) (b)

Fig. 4. Recognition rates for the 15 experiments. (a) Methods of set matching. (b) Methods of set matching combined with discriminative transformations. (The variation between the training and test data of the experiments increases along the horizontal axis. Note that (a) and (b) have different scales for vertical axis.)

Second, methods combined with any discriminant function were compared in Figure 4 (b). Note that Figure 4 (a) and (b) have different scales. By taking MSM as a gauging proxy, 1-NN, 10-NN and Hausdorff-distance in the LDA subspace and CMSM were compared with the proposed algorithm. Here again, 10-NN was the best of the three LDA methods. For better visualization of comparative results, the performance of 1-NN and Hausdorff in LDA was removed from the figure. 10-NN-LDA yielded a big improvement over 10-NN-PCA but the accuracy of the method again greatly varied across the experiments. Note that 10-NN-LDA outperformed MSM for similar conditions between the training and test sets, but it became noticeably inferior as the conditions changed. The recognition rate of NN-LDA was considerably inferior to our method for the more difficult experiments (experiments 11 to 15 in Figure 4 (b) where lighting conditions differ significantly). The accuracy of our method remained high at 94%. Note that the experiments 11 to 15 in Figure 4 are more realistic than the first half because they have greater variation in lighting conditions between training and testing. The proposed method also constantly provided a significant improvement over MSM by 5-10% reaching almost more than 97% recognition rate. Compared with CMSM, the proposed method achieved both higher accuracy and efficiency. CMSM has to be optimized aposteriori by dimensionality selection. By contrast, MMCC does not need any feature selection.

4.2 Experiment on Generic Object Recognition

The ALOI database [17] with 500 general object categories of different viewing angles provides another experimental data set for the comparison. The training and test sets were set up with different viewing angles of the objects, see Figure 5 (a). Object images were segmented from the simple background and scaled to 20×20

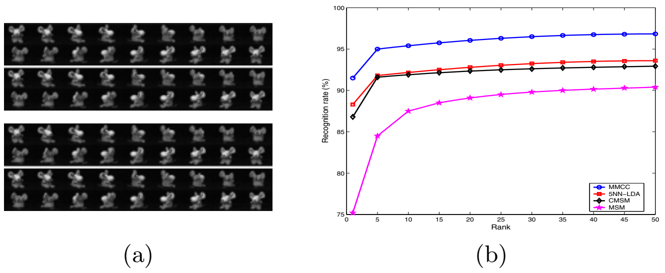


Fig. 5. ALOI experiment. (a) top two rows: the training set of one object; bottom two rows: two test sets. (b) Cumulative identification plots of several methods.

pixel size. The images for each object is divided into four groups, with viewing angle distributed by {0, 20, 40, ... 340}, {5, 25, 45, ... 345}, {10, 30, 50, ... 350} and {15, 35, 55, ... 355}. Two sets are for training and the left two for test. The methods of MSM, NN-LDA and CMSM were compared with the proposed method in terms of identification rate. The PCA dimensionality of each set was fixed to 5 and thus 5 canonical correlations were exploited for MSM, CMSM and the proposed method. Similarly, 5 nearest neighbors were used in LDA. See Figure 5 (b) for the cumulative identification rates. Unlike the face experiment, NN-LDA yielded better accuracy than both MSM and CMSM. This might be due to the nearest neighbours of the training and test set differed only slightly by the five degree pose difference (The two sets had no changes in lighting and they had accurate localization of the objects.). Here again, the proposed method were substantially superior to MSM, CMSM and NN-LDA.

5 Conclusions

This paper presents a margin maximizing discriminant analysis of canonical correlations for multiple-shot based object recognition. Our goal is to derive a mapping so that, in the mapped space, the canonical correlations of the intra-class image sets can be minimized and canonical correlations of the inter-class image sets can be maximized. This is done by learning a margin-maximized discriminant function of the canonical correlations. The proposed method is tested on two large data sets for two object recognition problems, face and generic object recognition. Experiment results shows its superior performance.

References

[1] G. Shakhnarovich, J.W. Fisher and T. Darrel, Face recognition from long-term observations. in: Proc. of ECCV, (2002) 851-868.

[2] O. Arandjelovic, G. Shakhnarovich, J. Fisher, R. Cipolla and T. Darrell, Face recognition with image sets using manifold density divergence, in: Proc. of CVPR, (2005).

[3] P.N. Belhumeur, J.P. Hespanha and D.J. Kriegman, Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection, PAMI, 19(1997)711-720.
[4] R. Huang, Q. Liu, H. Lu and S. Ma, Solving the Small Sample Size Problem of LDA, in: Proc. of ICPR, (2002).
[5] L.F. Chen, H.Y.M. Liao, J.C. Lin, M.D. Kao and G.J. Yu, A New LDA-Based Face Recognition System which Can Solve the Small Sample Size Problem, Pattern Recognition, 33(2000)1713-1726.
[6] H. Yu and J. Yang, A Direct LDA Algorithm for High-Dimensional Data With Application to Face Recognition, Pattern Recognition, 34(2001)2067-2070.
[7] S. Satoh, Comparative evaluation of face sequence matching for content-based video access, in: Proc. of FGR, (2000).
[8] M. Bressan and J. Vitria, Nonparametric discriminant analysis and nearest neighbor classification, Pattern Recognition Letters, 24(2003)2743-2749.
[9] O. Yamaguchi, K. Fukui and K. Maeda, Face recognition using temporal image sequence, in: Proc. of FGR, (1998)318-323.
[10] L. Wolf and A. Shashua, Learning over sets using kernel principal angles, Journal of Machine Learning Research, 4(2003)913-931.
[11] K. Fukui and O. Yamaguchi, Face recognition using multi-viewpoint patterns for robot vision, in: Proceedings of International Symposium of Robotics Research, (2003).
[12] H. Hotelling, Relations between two sets of variates, Biometrika, 28(1936)321-372.
[13] A. Bjorck and G.H. Golub, Numerical methods for computing angles between linear subspaces, Mathematics of Computation, 27(1973)579-594.
[14] K. Lee, M. Yang and D. Kriegman, Video-based face recognition using probabilistic appearance manifolds, in: Proc. of CVPR, (2003)313-320.
[15] S. Zhou, V. Krueger and R. Chellappa, Probabilistic recognition of human faces from video, CVIU, 91(2003):214-245.
[16] P. Viola and M. Jones, Robust real-time face detection, IJCV, 57(2004)137-154.
[17] J.M. Geusebroek, G.J. Burghouts and A.W.M. Smeulders, The Amsterdam library of object images, IJCV, 61(2005)103-112.
[18] E. Oja, Subspace Methods of Pattern Recognition. (Research Studies Press, 1983)
[19] H. Li, T. Jiang and K. Zhang, Efficient and Robust Feature Extraction by Maximum Margin Criterion, in: Proc. of NIPS 16, (2004).
[20] K. Fukunnaga, Introduction to Statistical Pattern Recognition. (Academic Press, 1991)
[21] Toshiba. Facepass. http://www.toshiba.co.jp/rdc/mmlab/tech/w31e.htm

A Novel 3D Statistical Shape Model for Segmentation of Medical Images

Zheen Zhao[1,2] and Eam Khwang Teoh[2]

[1] Department of Psychiatry, Duke University, Durham, NC, USA
kurtzhao@pmail.ntu.edu.sg
[2] School of Electrical & Electronics Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798
ekteoh@ntu.edu.sg

Abstract. A 3D Partitioned Active Shape Model (PASM) is proposed in this paper to address the problems of 3D Active Shape Models (ASM) caused by the limited numbers of training samples, which is usually the case in 3D segmentation. When training sets are small, 3D ASMs tend to be restrictive, because the plausible area/allowable region spanned by relatively few eigenvectors cannot capture the full range of shape variability. 3D PASMs overcome this limitation by using a partitioned representation of the ASM. Given a Point Distribution Model (PDM), the mean mesh is partitioned into a group of small tiles. The statistical priors of tiles are estimated by applying Principal Component Analysis to each tile to constrain corresponding tiles during deformation. To avoid the inconsistency of shapes between tiles, samples are projected as curves in one hyperspace, instead of point clouds in several hyperspaces. The deformed model points are then fitted into the allowable region of the model by using a curve alignment scheme. The experiments on 3D human brain MRIs show that when the numbers of the training samples are limited, the 3D PASMs significantly improve the segmentation results as compared to 3D ASMs and 3D Hierarchical ASMs, which are the extension of the 2D Hierarchical ASM to the 3D case.

1 Introduction

Deformable models are initially proposed in [1], known as "Snakes", which impose a constraint of local smoothness derived by elastic forces. However, snakes are often too flexible, because they tolerate deformation as long as the deformation is smooth. When models are attracted to spurious edges, the models are not likely recover this drift, unless this deformation creates sharp peaks. Thus, the robustness to rough initial positions suffers. Therefore, snakes are more suitable for human-machine interfaces.

On the other hand, statistical shape models, which constrain the deformation using statistical priors derived from a set of training samples, overcome these limitations. Among statistical shape models, the 2D Active Shape Model (ASM) [2] provide a promise to improve the robustness of the model to local minimum by restricting the deformation within the allowable region. Hill *et al.* extend the

2D ASM to the 3D case for 3D medical image analysis in [3]. Paulsen *et al.* [4] build a 3D ASM for testing of gender related difference in size and shape of the ear canal. Kaus *et al.* [5] construct 3D ASMs to segment vertebras and femurs from CT images.

However, in 3D segmentation, 3D ASM often constrains itself from catching details during deformations. This is because the number of eigenvectors/eigenmodes cannot exceed the number of training samples, in the case of the dimension of the model being large, which is usually the case in 3D segmentation, as the dimension of the model is typically two or three orders of magnitude higher than the number of training samples. It is difficult to estimate a high-dimensional probability distribution from a small training set. Therefore, the allowable region spanned by the relatively few eigenvectors limits the deformation of ASM to catch details. A solution is by using large training sets. However, it requires manually segmenting 3D images slice by slice, which is very laborious. Therefore it is inconvenient to build a large training set.

There have been various attempts to address this limitation. ASM combined with elastic models has been proposed in [6], but the detailed deformations are regulated by elastic forces, which do not reflect true shape variability. In [7], Davatzikos *et al.* proposed 2D Hierarchical Active Shape Model (HASM), a hierarchical scheme based on 2D curve partition and 2D curve wavelet decomposition to keep details. Their experiments have shown promising results in this "short of samples " scenario. However, in [7], the spatially partitioned representations of objects, both curve segmentation and wavelet decomposition, introduce shape inconsistency. Because in [7], the allowable regions/plausible areas of bands are independent to each other, illegal shapes could be tolerated during model fitting. As a result, this model could be vulnerable to noises and low contrasts. The authors hierarchical representation and a deformation scheme to subdue this problem, however the approaches only partially solve this problem.

In this paper, a 3D Partitioned Active Shape Model, which uses curve alignment to fit models during deformations, is proposed. In 3D PASM, each training sample and deformed model is represented as a curve, instead of a point as ASM and HASM. After the model has deformed to find the best match in the image, the deformed model is projected back. The curve representing the deformed model is aligned with the closest training sample by affine transformation. Furthermore, to investigate the strength and weakness of 3D PASM, the 2D HASM in [7] is extended to 3D HASM in this work based on the mesh partitioning algorithm proposed in this paper.

2 3D Partitioned Active Shape Models

A 3D Partitioned Active Shape Model is described in this section. It is assumed the vertices of manual segmentations have been corresponded to construct a Point Distribution Model (PDM). It means that N training samples are available as sets of K corresponding landmarks in the 3D space.

2.1 Construction of Allowable Regions

In conventional ASMs, the training samples are aligned using Procrustes alignment. The vectors $X_n, \ n = 1, \ldots, N$ are then formed by concatenating the coordinates of the K landmark points of N samples. Therefore, the dimension of X is $3K \times N$. To reduce the dimensionality of X, Principal Component Analysis (PCA) is applied to X. The eigenvectors $e_1, \ldots, e_{N-1}$, which are corresponding to the nonzero eigenvalues of the covariance matrix of X, are calculated. In the 3D case, it is typical that $N \ll 3K$. Therefore it is likely that S, spanned by $e_1, \ldots, e_{N-1}$, cannot include the full range of shape variation. As a result, 3D ASM, which projects the tentative shape by using S, tends to reconstruct a shape without fine details.

3D PASM solves this problem by a partitioned representation of 3D ASMs. Because a 3D ASM is represented as a mean mesh and its eigen variations of individual tiles, the partitioned representation of the ASM is the partitioned mean mesh and eigen variations of tiles. Mesh partition means that a mesh is partitioned into a group of surface patches, which are called tiles in this study.

If only the statistical prior of a small tile, which comprises of $K^{'}$ vertices, of the whole mesh is required, N samples will be adequate to include the variation of this tile, as long as the $K^{'}$ is small enough. Thus fine details of this tile could be captured in the allowable region.

Therefore, the mesh is partitioned into tiles so that the tiles cover all faces of the mesh and each tile consists of roughly $K^{'}$ vertices. The PCA is then independently applied to each tile to form the statistical prior for each tile.

It is assumed that the model has been partitioned into M tiles. For the j^{th} tile, X_i^j is the vector obtained by concatenating the coordinates of vertices of the j^{th} tile. The prior of the j^{th} tile is formed by:

Step 1. Compute the mean of the data,

$$\bar{X^j} = \frac{1}{N}\sum_{i=1}^{N} X_i^j \tag{1}$$

Step 2. Compute the covariance of the data,

$$S^j = \frac{1}{N-1}\sum_{i=1}^{N}(X_i^j - \bar{X^j})(X_i^j - \bar{X^j})^T \tag{2}$$

Step 3. Compute the eigenvectors, ϕ_i^j and corresponding eigenvalues λ_i^j of S (sorted so that $\lambda_i^j \geq \lambda_{i+1}^j$).

If Φ^j contains the t^j eigenvectors corresponding to the largest eigenvalues, then any of the training set can approximate X by using

$$X^j \approx \bar{X^j} + \Phi^j b^j \tag{3}$$

where $\Phi^j = (\phi_1^j|\phi_2^j|\dots|\phi_{t^j}^j)$ and b^j is a t^j dimensional vector given by

$$b^j = {\Phi^j}^T (X^j - \bar{X}^j) \tag{4}$$

The vector b^j defines a set of parameters of a deformable model. By varying the elements of b^j we can vary the shape X^j using Eq. 3.

Therefore, when the model is attracted to their best match, the deformed model of j^{th} tile is Y^j, which can be approximated by the parameters b_Y^j

$$b_Y^j = {\Phi^j}^T (Y^j - \bar{X}^j) \tag{5}$$

On the contrary, ASM represents the shape of the whole model by

$$b_Y = \Phi^T (Y - \bar{X}) \tag{6}$$

2.2 Mesh Partitioning

In this section, an algorithm is described to partition a mesh M into a group of tiles. A result of applying this algorithm is shown in Fig.1. This algorithm segments mesh M to tiles $\tau_1, \dots, \tau_s$, which cover all faces of M, given a set of sites positioned at the centroids of the *site faces* $S = f_1, \dots, f_s$. A face of M is randomly selected as the initial site. Once the tile associated with the site stops growing, another face of M from faces not covered by grown tiles is selected as the next site. The procedure is repeated until all faces of M are covered by tiles.

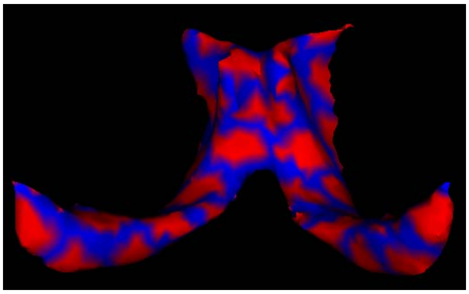


Fig. 1. A partitioned mesh. Tiles are shown in red; boundaries are shown in blue.

A tile τ_i is a collection of faces for which the closest site face is f_i. The measure of distance between faces is an approximation of geodesic distance over the mesh. It is defined by constructing a dual directed graph of the mesh M *i.e.* the nodes of the graph correspond to faces of M, and the edges of the graph connect nodes of adjacent faces. The cost of edges in this directed graph is set to the distance between centroids of the corresponding faces. This distance is defined as the length of the shortest path in this directed graph.

Constructing a tile is a single-source shortest path problem in the graph, which is solved by a variant of Dijkstra's algorithm . The algorithm grows a tile until the size of the tile reaches the maximum. The size of a tile is defined as the number of vertices included in the faces of the tile. The maximum is set so that the number of tiles is not much bigger than the sizes of individual tiles.

2.3 Model Fitting Scheme

In ASM, after applying PCA, each training sample is represented as one point in a hyperspace. The training set constructs a cloud in this space. The allowable region can then be estimated from the distribution of the cloud of points/samples.

When an ASM is partitioned into H tiles, PCA is applied to the H tiles separately, instead of the whole model. Therefore, the training samples are represented as point clouds in the H hyperspaces, which are generated by PCA. If they are combined independently, like traditional HASM in [7], shape inconsistency could be introduced because the shape relevancy between tiles are not considered.

To avoid this inconsistency, in 3D PASM, these hyperspaces are combined to one hyperspace. Samples and the deformed models are represented as curves in this single hyperspace, instead of point clouds in individual hyperspaces. A curve alignment scheme is used in model fitting to preclude inconsistent/illegal shapes during deformations. The detailed description goes as follows.

After PCA is independently applied to the coordinates of vertices of each tile, each training sample is then represented as points in H hyperspaces, one point in each hyperspace. To combine the hyperspaces, the indices of tiles are introduced as another dimension, the H hyperspaces are then combined to one hyperspace. As a result, each training sample is represented as a curve in the new hyperspace, instead of a point.

During a deformation, after the model points move to find the best match in a test image, model points are projected back to the hyperspace as a curve, called model curve. The closest training sample, in terms of Euclidean distance, is chosen as the target. The model curve is then transformed to align with the target by affine transformation. The affine invariant alignment can be estimated by using the Least Square method in [8].

The model fitting scheme of 3D PASM is shown in the left figure in Fig. 2, where solid lines indicate models, the broken lines stand for training samples. Assume that the model/mesh is partitioned into 3 tiles, and the dimensionality of the 3 hyperspaces is 1D after individual PCA applications. The indices of the tiles are introduced as another dimension. The two training samples are then represented by two curves. To fit the deformed model points, the curve representing the deformed model is aligned with the curve representing training sample 1 by using affine transformation, which is the closest to the model curve. As a result, the shape of the fitted model could not be far from the closest sample. The inconsistency between tiles is avoided, because the relevancy between shapes of tiles from the samples are enforced. On the contrary, the traditional HASM [7], which is shown in the right figure in Fig. 2, treats training samples as points in individual tiles. The deformed model points are fitted to the model by truncation, which could tolerate shape inconsistency, because the shape of the fitted model could be different from any training sample.

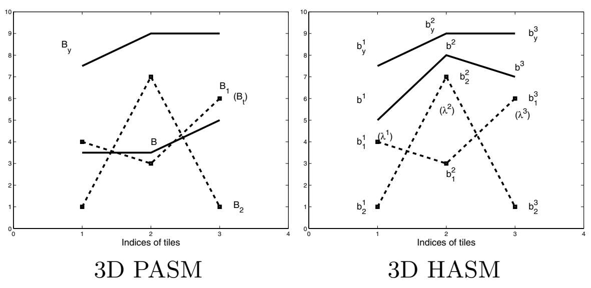


Fig. 2. The model fitting schemes of the 3D PASM and the 3D HASM

3 Experimental Results

In this section, the experimental results and quantitative analysis of the performance of 3D ASM, 3D PASM and 3D HASM are presented. The 2D HASM in [7] is extended to 3D HASM based on mesh partitioning in this study to compare with 3D PASM.

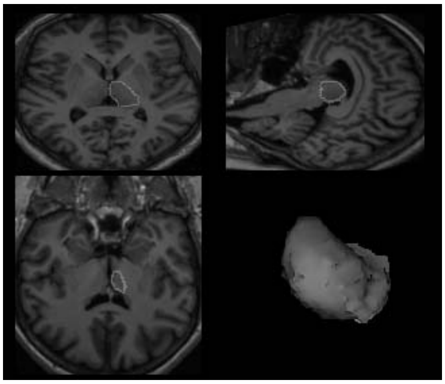
3.1 Image Data

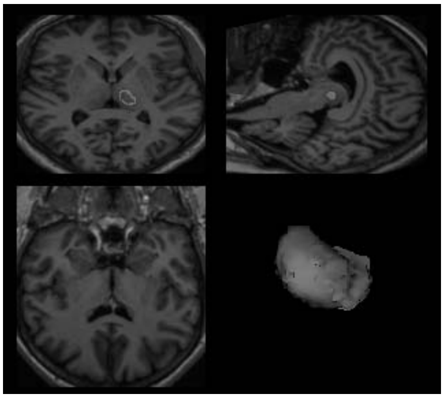
The training sets and images used in this work are from the Internet Brain Segmentation Repository (IBSR) [9]. The first set is IBSR v1.0. They are 20 normal T1-weighted MR brain 3D images and their manual segmentations. The second set is IBSR v2.0. They are 18 normal T1-weighted MR brain 3D images and their manual segmentations. The objects to segment are Lateral Ventricles extracted from IBSR v1.0, Lateral Ventricles extracted from IBSR v2.0, Left Thalamus Propers extracted from IBSR v2.0 and Left Hippocampuses extracted from IBSR v2.0. For simplicity, they are called LV1, LV2, LTP and LH in this study, respectively.

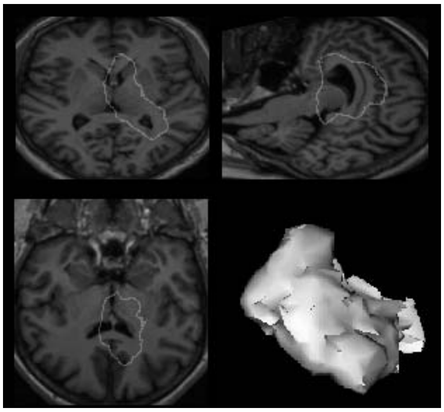
These four objects and two sets of images are chosen because of their diversity. This diversity is desired to investigate the performance of 3D PASM. Images of IBSR v1.0 are noisy, because they were acquired a few years ago. Images of IBSR v2.0 have higher quality and are free from noise. The contrast of the region around lateral ventricles is sharp. On the contrary, the contrast of the region around thalamus propers and hippocampuses is relatively vague, which means that it is more difficult to segment thalamus propers and hippocampuses than lateral ventricles.

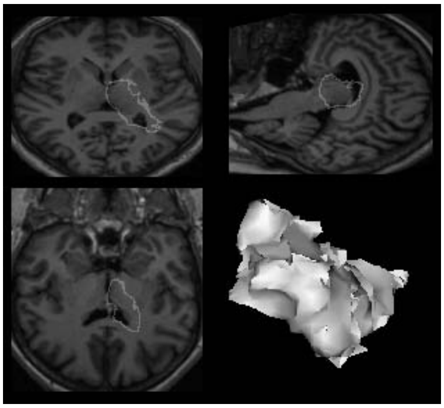
3.2 A Comparative Study of Object Segmentation Using Different Deformable Models

Standard 3D ASM, 3D HASM and 3D PASM are applied to segment LV1, LV2, LTP and LH from test images. For comparison purpose, the initialization and parameters of the three models are exactly the same, wherever applicable. The


(a) Manual Segmentation


(b) Initialization: 5.5668


(c) ASM: 10.2965


(d) 3D HASM: 5.4502

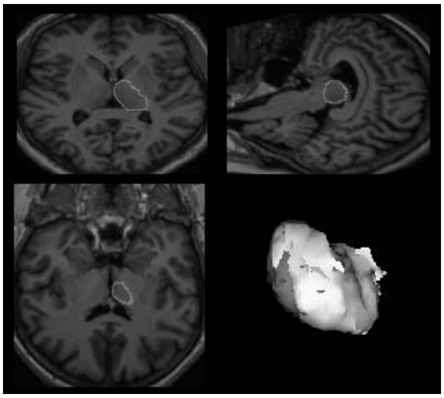

(e) 3D PASM:1.1381

Fig. 3. The comparative study on the accuracy of segmentations for LTP by using different models

Table 1. The average segmentation errors of the comparative study of object segmentation using different deformable models(in voxels)

	LV1	LV2	LTP	LH
3D ASM	2.6364	1.5470	4.6983	9.1490
3D HASM	1.6282	1.5609	2.6014	4.4104
3D PASM	1.2469	1.5104	1.1717	4.2943

"leave-one-out" method is used in this experiment. The "segmentation error" is used to measure the accuracy of the segmentations, which is defined as the distances between the segmentation results and the corresponding manual segmentations. This metric is defined in [10,11]. If a segmentation error is bigger than 1/3 of initialization error, the segmentation is defined as a "failure" in this study.

For each of the four cases, one example of the manual segmentation, initialization and the segmentation results of models and the quantitative results of segmentations are illustrated in Fig. 3. The segmentation by models are overlayed with cross-sections (in gray contours) and the 3D surface of manual segmentations, where the dark surface is the segmentation by the models and the bright one is the manual segmentation. Their average is listed in Table 1. The four figures show the segmentation of objects with high contrast in noisy images, of objects with low contrast and simple shapes, of objects with low contrast and complex shapes, and of objects with high contrast in non-noisy images.

In several cases, the standard 3D ASMs failed to find the desired boundaries, because 19 or 17 samples are not sufficient for ASMs to accurately estimate the distribution of samples. During iterations, the rigidity of ASMs tends to keep model points from moving to approximate appropriate shape details. The accumulation of errors in iterations could lead to segmentation failures. Furthermore, model points have difficulty to find their best match in noisy images. As a result, during iterations, when a few points find their best matches, the shape of the model cannot change accordingly to guide other points, because 3D ASMs are too restrictive.

3D HASMs have partitioned representation, so they do not have this "rigidity" problem. But they still failed to accurately segment in a few cases because they tolerate illegal deformations, *i.e.* their allowable regions are not plausible. On the contrary, the 3D PASM improved the segmentation accuracy significantly because of the partitioned representation and the model fitting scheme using curve alignment. The improvements by 3D PASMs on LV2 are not as significant as other objects because of the images of LV2 are very clear and the lateral ventricles have strong contrasts. However, if the images are noisy or the contrasts are low, 3D PASM brings significant improvements over 3D ASM and 3D HASM.

3.3 On System Sensitivity Study

To investigate the reproducibility of the PASMs, *i.e.*, the sensitivity of the final segmentation result to the initialization, a study of sensitivity is conducted. The leave-one-out experiments are carried out 5 times on LV1. The initialization position are different for each experiment. The mean and standard deviation of segmentation errors and initialization are listed in Table 2.From the table, when the initialization position changes radically, reflected by the range of the initialization errors, the performance of the segmentations using PASM is relatively stable.

Table 2. The mean and STD of final segmentations when initialized with different initialization error (in voxels).

Indices of experiments	Initialization		Segmentation	
	Mean	*STD*	*Mean*	*STD*
No. 1	2.1046	0.4440	1.3307	0.7642
No. 2	8.7399	0.6415	1.2469	0.6165
No. 3	8.8850	0.1030	1.4036	0.8315
No. 4	15.5480	1.1654	1.2701	0.6598
No. 5	17.4330	0.0515	1.4670	0.8765

4 Conclusion and Discussion

The segmentation using small training sets is an important problem. The challenge is that the statistical prior of high-dimensional features cannot be accurately estimated from small training sets. However, this estimation is accurate if the dimensionality of the features is low. A 3D Partitioned Active Shape Model is proposed in this work to solve this "short of training samples" problem. 3D PASM uses the partitioned representations to decrease the dimensionality of the features. To avoid the inconsistency of shapes caused by the partitioned representation, the relationship of the shapes of the tiles is taken into consideration. 3D PASMs are applied to segment structures from 3D human brain MRIs. The results show that when the number of the training samples is limited, 3D PASMs substantially improve the segmentation results as compared to 3D HASMs and 3D ASMs.

A concern is that one vertex in a tile may actually be directly connected to another vertex in another tile and the motion is coupled while deforming the mesh. Actually it is why the tiles are projected and matched as curves instead of points. By aligning the curves, the motion among tiles is kept. Regarding the question about the nature of the boundaries between the tiles, they are overlapping vertices because the tiles are composed of sites/faces instead of vertices. The reason for setting the cost of the edges to the distance between centroids is to generate tiles with similar sizes.

References

1. Kass, M., Witkin, A.: Snake: Active contour models. International Journal of Computer Vision **1** (1988) 321–331
2. Cootes, T., Hill, A., Taylor, C., Haslam, J.: The use of active shape models for locating structures in medical images. Image and Vision Computing **12** (1994) 355–366

3. Hill, A., Thornham, A., Taylor, C.J.: Model-based interpretation of 3D medical images. In Illingworth, J., ed.: Proc. of The British Machine Vision Conference 1993. (1993) 339–348
4. Paulsen, R.R., Larsen, R., Laugesen, S., Nielsen, C., Ersbøll, B.K.: Building and testing a statistical shape model of the human ear canal. In: Proc. Medical Image Computing and Computer-Assisted Intervention - MICCAI 2002, 5th Int. Conference, Tokyo, Japan, Springer. (2002) 373–380
5. Kaus, M.R., Pekar, V., Lorenz, C., Truyen, R., Lobregt, S., Weese, J.: Automated 3-D PDM construction from segmented images using deformable models. IEEE Transactions on Medical Imaging **22(8)** (August 2003) 1005 –1013
6. Cootes, T.F., Taylor, C.J. In: Statistical Models of Appearance for Computer Vision. Technical Report, University of Manchester (2001)
7. Davatzikos, C., Tao, X., Shen, D.: Hierarchical active shape models, using the wavelet transform. IEEE Transactions on Medical Imaging **22(3)** (March 2003) 414–423
8. Ip, H.H.S., Shen, D.: An affine-invariant active contour model (ai-snake) for model-based segmentation. Image and Vision Computing **16 No.2** (1998) 125–146
9. IBSR: The internet brain segmentation repository (ibsr). (avaliable at http://www.cma.mgh.harvard.edu/ibsr/)
10. Zhao, Z., Teoh, E.K.: A novel framework for automated 3D PDM construction using deformable models. In: Proceedings of the SPIE, Vol. 5747, pp. 303-314, SPIE Medical Imaging, San Diego, CA (2005. Avalialbe at www.duke.edu/∼kurtzhao/ZhaoSpie.pdf)
11. Zhao, Z. In: A Novel 3D Statistical Shape Model for Segmentation of Medical Images. PhD Thesis Nanyang Technological University, Singapore (2006)

Scale Consistent Image Completion

Michal Holtzman-Gazit and Irad Yavneh

Computer Science Department
Technion–I.I.T. Technion City, Haifa 32000, Israel
{mikih, irad}@cs.technion.ac.il

Abstract. Most patch based image completion algorithms fill in missing parts of images by copying patches from the known part of the image into the unknown part. The criterion for preferring one patch over another is the compatibility or consistency of the proposed patch with the nearby region that is known or already completed. In this paper we propose adding another dimension to this consistency criterion, namely, scale. Thus, the preferred patch is chosen by evaluating its consistency with respect to smoothed (less detailed) versions of the image, as well as its surroundings in the current version. Applied recursively, this approach results in a multi-scale framework that is shown to yield a dramatic improvement in the robustness of a good existing image completion algorithm.

1 Introduction

The new age of digital images allows us to take a picture and then alter it by removing an undesired object it contains. The question is then how to complete the missing information so that the image still looks natural. In recent years many new algorithms for this problem have been proposed, and yet the problem remains difficult.

The problem is difficult largely because the term "looks natural" is hard to define mathematically. This difficulty is closely tied with the multi-scale nature of images, especially images containing complex textures. The aim of this work is to develop a systematic approach for addressing the latter property. The main idea is that the completed image must look natural at *all* (suitably defined) scales of the image. This implies a certain consistency between a completed image, and similarly completed "smoother" versions of the image containing less and less texture details. In effect, an additional dimension—scale—is thus encompassed in the completion process. Employing a given completion method within a suitable multi-scale framework, that imposes this consistency, should therefore be expected to improve its robustness substantially. We demonstrate this fact in a set of experiments with synthetic and natural images.

Recent studies on image completion may be divided into a few main categories. Inpainting methods are designed to repair images by removing small artifacts such as scratches, small "holes" or overlaid text. These include PDE based methods [1,2,3], Fast marching [4], diffusion by convolution [5], and other more complicated methods such as [6,7,8]. The main drawback of all the above methods is that, in large holes, the data inside the hole is smoothed out. Therefore, they are suitable mostly for small artifacts.

A second approach is using texture synthesis in order to fill large holes with pure texture, by sampling the texture and generating a large area with the same texture.

In [9,10,11,12,13] a new texture is synthesized pixel by pixel, by looking for similar neighborhoods in the example texture. Other methods [14,15,16] copy full patches (also called "blocks") of different sizes and shapes from the source image to the target.

Recently, more complex methods have been designed, mainly for object removal and completion of the complex textures behind it. These variations include different orders of filling [11,17], segmentation [18], image decomposition [19], rotation and scaling [20], Gaussian pyramids [21], global consistency measures [22], and user guidance [23].

The method we shall be using in our experiments is the algorithm proposed by Criminisi et al. in [24]. This is an effective yet computationally efficient approach for patch based completion. The method uses exemplar based synthesis, where the order of the filling is determined by the direction and sharpness of gradients impinging on the boundary of the missing part of the image. Thus, linear structures in the image tend to be continued into the missing region. Although it is relatively simple, it performs well for many examples.

Our aim is to show how to improve the robustness of image completion procedures by incorporating them in a multiscale framework. In section 2 we present the main ideas and underlying assumptions in an abstract form and propose an approach for implementing them. Section 3 describes a specific implementation, along with a detailed description of a completion algorithm. Section 4 present experimental results, and section 5 is a summary and conclusion.

2 Scale-Consistency

2.1 The Main Ideas and Notation

Let $I = I(\Omega) : \Omega \rightarrow [0,1]^{d\times|\Omega|}$ denote an image on a set of pixels, Ω. Here, e.g., $d = 1$ for grey-level images and $d = 3$ for color images. The fact that grey levels and colors are quantized in practice is unimportant for this discussion. Assume that Ω is partitioned into two regions: $\Omega = \Omega_k \cup \Omega_m$, where Ω_k is the subset of pixels where I is known, while Ω_m is the region where I is missing; see illustration in Fig. 1. Throughout this paper, missing pixels will be set to 0^d (black). An image completion algorithm is a function, $C : [0,1]^{d\times|\Omega|} \rightarrow [0,1]^{d\times|\Omega|}$, such that $\mathbf{I} = C(I(\Omega))$ satisfies $\mathbf{I}(\Omega_k) = I(\Omega_k)$. That is, C returns an image that is identical to I wherever the latter is known. The purpose of C is to introduce values in $\mathbf{I}(\Omega_m)$ such that $\mathbf{I}(\Omega)$ "looks natural". Evidently, this is a subjective term, which is, to a large extent, why this problem is difficult. Nevertheless, let us suppose that we are in possession of a quality measure, Q, such that $Q[\mathbf{I}_1] > Q[\mathbf{I}_2]$ implies that $\mathbf{I}_1$ is a higher-quality completion than $\mathbf{I}_2$. In fact, we assume that we can even compare the qualities of completions of different input images.

Next, we introduce the notion of *smoothing*. A smoothing algorithm is a function, $S : [0,1]^{d\times|\Omega|} \rightarrow [0,1]^{d\times|\Omega|}$, such that $I_S = S(I)$ is a less-detailed version of I retaining the same set of missing values, Ω_m, which may or may not be empty. A simple example is a convolution with a Gaussian (modified such that missing values remain black), though we are more interested in smoothing algorithms that preserve important

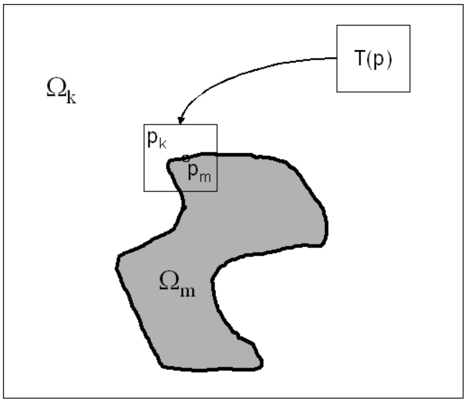


Fig. 1. An illustration of the known and missing regions and the source and target patches

features, particularly edges. Below we shall be using nonlinear diffusion smoothers for this, but other methods may also be used, such as the hierarchical segmentation of [25].

The key idea underlying our approach is that a high-quality completion can only be achieved if the smoothing, S, and completion, C, approximately commute:

$$C\left(S(I)\right) \approx S\left(C(I)\right) \,. \tag{1}$$

This assumption is motivated by the fact that both sides of (1) represent images that are smooth versions of a naturally completed I. It compares between the completions of the fine-detailed image, I, and that of its smooth version, I_S. If (1) is satisfied, we say that the completion is *scale-consistent.*

Additionally, we will assume that a smoothed image is easier to complete well than its more detailed version:

$$Q[S(I)] \geq Q[I] \,. \tag{2}$$

This is a well-established notion that has been exploited for image completion, e.g., in [6,19]. On the basis of these observations, we next describe a general multi-scale patch-based image completion approach.

2.2 A Patch-Based Scale-Consistent Completion

We restrict our discussion to patch-based completion methods of the following type. The input is an image, $I(\Omega)$, with values missing (blacked out) in $\Omega_m \subset \Omega$, but known in the complement, Ω_k. The output image, $\mathbf{I}(\Omega)$, is initialized by setting $\mathbf{I} = I$. At each step of the algorithm, we consider a subset of pixels called a *target patch*, $p \subset \Omega$, which overlaps with both the known (or already completed) and missing parts of the image: $p_m \equiv p \cap \Omega_m \neq \emptyset$, and $p_k \equiv p \setminus p_m \neq \emptyset$. Usually, the shape and size of p are pre-defined, and the shape is simple, e.g., a square or a disk of pixels. Next, a *source patch*, $T(p) \subset \Omega_k$ is selected, where T is one of a family of simple transformations that preserve shape and size—usually just translations, but possibly also some rotations. Then, the missing portion of the image corresponding to p_m is completed by setting:

$\mathbf{I}(p_m) \leftarrow \mathbf{I}(T(p_m))$. Finally, the missing part is reduced by redefining $\Omega_m \leftarrow \Omega_m \backslash p_m$. This completes one step of the algorithm. The process is repeated for a new target patch, etc., until $\Omega_m = \emptyset$.

Two key decisions are the choices of p and the corresponding $T(p)$ at each step. Given a target patch, p, the source patch, $T(p)$, is generally chosen such that $\mathbf{I}(T(p_k))$ is "as similar as possible" to $\mathbf{I}(p_k)$, employing some suitable measure of similarity. We write this criterion as:

$$\mathbf{I}(T(p_k)) \approx \mathbf{I}(p_k)\,. \tag{3}$$

Different algorithms use different measures of similarity, different allowable sets of transformations, and/or additional considerations such as preferring "common" patches over more "exotic" ones, even if the match is not as good. The point here is that, very often, there are several possible target patches of comparable quality by a given measure, and the fact that one of these happens to be slightly better than the others is not very compelling compared to other considerations.

The choice of p at each step is also important. Most algorithms choose a patch whose intersection with Ω_m is relatively small (so that p_k is sufficiently rich for gauging the suitability of $T(p)$ via (3).) Another important consideration is the details of the image in the vicinity Ω_m. For example, preference may be given to a region with a strong edge impinging on the boundary of Ω_m.

Next we describe the scale-consistent approach, using just two levels of detail for simplicity. Later, we generalize to a multi-scale framework. Given a patch-based completion method, C, and a smoothing function S, denote $I_S = S(I)$, and $\mathbf{I}_S = C(I_S)$. For the latter we use, of course, criterion (3), applied to $\mathbf{I}_S$, i.e.,

$$\mathbf{I}_S(T_S(p_k)) \approx \mathbf{I}_S(p_k)\,, \tag{4}$$

where $T_S(p_k)$ is the source patch selected in the smoothed image.

We would like to complete I while respecting our key underlying assumption, (1). Since we do not yet know the completed image at this stage, and therefore cannot evaluate $S(\mathbf{I}(T(p)))$, which is required for the right-hand side of (1), we approximate it by $\mathbf{I}_S(T(p))$. This leads to the scale-consistency criterion,

$$\mathbf{I}_S(T(p)) \approx \mathbf{I}_S(p)\,. \tag{5}$$

Equations (3,4,5) represent three criteria that we would like to satisfy simultaneously. The importance we attach to each will influence the reconstruction. In this work, we are motivated by (2) to give criterion (4) a higher precedence than the other two. Thus, we first complete the entire smoothed image, I_S, using C, without considering scale consistency. Only then do we complete I, taking both (3) and (5) into account with approximately equal weight as described in the next section. Note that giving much greater weight to (3) would yield the usual completion, $C(I)$. On the other hand, giving much greater weight to (5) would mean setting $T = T_S$, which completely ignores the detailed image. By employing both criteria equally, we obtain a completion that is scale consistent, while still complying with the rules of C.

The approach of completing I_S first, and only later completing I in a scale-consistent manner, has several advantages. It is relatively simple, it is easy to generalize to a multi-scale framework (i.e., several levels of detail), and it also allows us to use target patches (possibly of different sizes) in different orders for $\mathbf{I}$ and $\mathbf{I}_S$. The disadvantage is that failure to obtain a high-quality completion of I_S will most likely result also in a similar failure for I. This might be overcome by allowing the fine-detailed image to also influence the completion of the smoothed image, e.g., by attaching a similar weight to the three similarity criteria. We leave this for future investigation.

3 A Specific SCIC Completion Algorithm

To implement the ideas proposed in the previous section, we generate a set of images $I_0, \ldots, I_n$ with varying levels of detail, using the smoothing function described in section 3.1. Here, $I_n = I$, the input image. We begin with the image containing the least detail, I_0, and complete the missing region using a patch based completion algorithm, C, which is a modification of the algorithm of [24] (see subsection 3.2). Next, we progressively complete the finer versions, ending with I_n. For each version but the smoothest—I_i for all $i > 0$—we construct a completion which is consistent with that of the previous level of detail, I_{i-1} (see subsection 3.3). The algorithm can be written as follows, with the details described below.

Algorithm 1. SCIC

```
Input: $I_0, \ldots, I_n = I$ images at varying levels of detail, and the missing region, $\Omega_m$ .
for detail level $i = 0, \ldots, n$ do
    $\Omega_m^i = \Omega_m$.
    while $\Omega_m^i \neq \emptyset$ do
        Select target patch, $p$ (see subsection 3.2).
        for each allowable patch size do
            Find the best matching source patch, $T(p)$ (see subsections 3.2 and 3.3).
        end for
        Choose the patch size and best match according to subsection 3.2.
        Set $\mathbf{I}(p_m) \leftarrow \mathbf{I}(T(p_m))$ .
        Set $\Omega_m^i \leftarrow \Omega_m^i \setminus p_m$.
    end while
end for
```

3.1 The Smoothing Function, S

We generate the progressively smoother images using Perona-Malik flow [26] for grey-level images, and Beltrami flow [27] for color images. These procedures apply an adaptive filter that preserves the strong edges in the image while progressively smoothing out texture. All the images generated are of the same size.

3.2 The Basic Completion Algorithm, C

Our basic completion algorithm is a modification of the algorithm of Criminisi et al. [24]. As described in section 2.2, the method is characterized by the choice of target patch, which determines the order in which the hole is filled, and the similarity measure, which determines the choice of source patch. We choose the target patch as in [24], by maximizing the product of two factors: the component of the image gradient tangential to the hole boundary, and the relative size of the known region within the target patch, $|p_k|/|p|$.

Next, we describe the modifications we introduce in order to improve the performance while also limiting, or even eliminating, user intervention.

Adaptive Patch Size. In [24] the size of p is fixed and predetermined by the user. This assumes that there is some single size that is suitable for very one of the completion steps. We prefer to avoid this assumption and limit user intervention. We therefore specify a set of allowable patch-sizes[1] that is independent of the image. At every completion step, each of the patch-sizes is tested, and the one giving the best match is chosen. The best match is determined by the weighted mean square similarity described below, but favoring larger patches over smaller ones, as the latter are more likely to introduce large-scale errors (e.g., by copying objects into the hole).

The search for the preferred block size employs an efficient hierarchical approach, such that the computational burden is not much greater than is required for a single patch size.

Similarity Measure. The choice of source patch, $T(p)$, is determined in [24] by the mean square of $\mathbf{I}(p_k) - \mathbf{I}(T(p_k))$, which we denote by $||\Delta(\mathbf{I}, p_k)||^2$. We modify the algorithm slightly by attaching greater weight to the differences in regions that are close to the border of p_m. The reason for this is that p_m will be replaced by $T(p_m)$, and a large difference along the border may create a visible "seam". The weight is efficiently calculated using a simple convolution with a kernel that varies with the distance from the border of p_m, which is relatively large at the seam but becomes constant a few pixels away from the seam.

3.3 The Scale Consistency Measure

We wish to satisfy (3) and (5) with approximately equal weight. Suppose that image I_{i-1} has already been completed. For each target patch, p, selected according to subsection 3.2, we search for the source patch, $T(p)$, that minimizes

$$C_1^{-1}||\Delta(\mathbf{I}_i, p_k)||^2 + C_2^{-1}||\Delta(\mathbf{I}_{i-1}, p)||^2 ,$$

where the norms are weighted as described above. Here, C_1 and C_2 are normalization constants, given by

$$C_1 = \sum ||\Delta(\mathbf{I}_i, p_k)||^2 , \quad C_2 = \sum ||\Delta(\mathbf{I}_{i-1}, p)||^2 ,$$

where the sum is taken over source patches for which $||\Delta(\mathbf{I}_i, p_k)||$ and $||\Delta(\mathbf{I}_{i-1}, p)||$ are not both dominating (hence, clearly inferior). This means that all source patches for which the Δ is larger than some other source patch in both images I_i and I_{i-1} are ignored when computing the constants C_1 and C_2, leaving only candidates that may turn out to be optimal. Including all the source patches in the normalization instead would yield irrelevant constants.

[1] All our patches, as in [24], are squares centered on pixels that lie on the border between the missing part of the image and the part that is known or already completed.

3.4 Computational Complexity

SCIC obviously introduces some computational overhead. Currently, for an search area of 400^2 and a missing region in the size of about 5000 pixels our non-optimized code runs for about 20 minutes on a 2.8 Gigahertz Pentium IV PC. We use a Matlab code with a C helper function for the search. However, with proper handling, the overall cost of completing the $n + 1$ images, $I_0, \ldots, I_n$, should be just a fraction more than only completing I_0. The lion's share of work invested in the completion is spent in searching for the best source patches, which is carried out over a very large number of different candidate patches in the known part of the image. The vast majority of these turn out to be poor choices. There is no reason to test these again when we complete images $I_1, \ldots, I_n$ (as the locations of the small percentage of "reasonable" candidates identified in image I_0 can be stored). Due to the scale consistency requirement, a patch that provides a poor match for the smooth image is extremely unlikely to be the best choice at a more detailed version of the image, and can therefore be ignored. This is easy to accomplish if we fill all the images in the same order and with patches of the same size, which would modify our algorithm somewhat. However, we can also retain our current algorithm but expand the search to include also patches that are in the neighborhood of source patches that gave reasonable results on the less detailed image. Still, we expect the total effort required for completing all but the smoothest image to be fairly modest compared with the cost of completing just I_0. A more precise assessment of overall cost using this approach is currently being investigated.

4 Experimental Results

We test the SCIC algorithm on both synthetic and natural images, and compare it to our implementation of the original algorithm of Criminisi et al. [24], and also, in part, with our modified version. Fig. 2 demonstrates the validity of assumption (2) and the efficacy

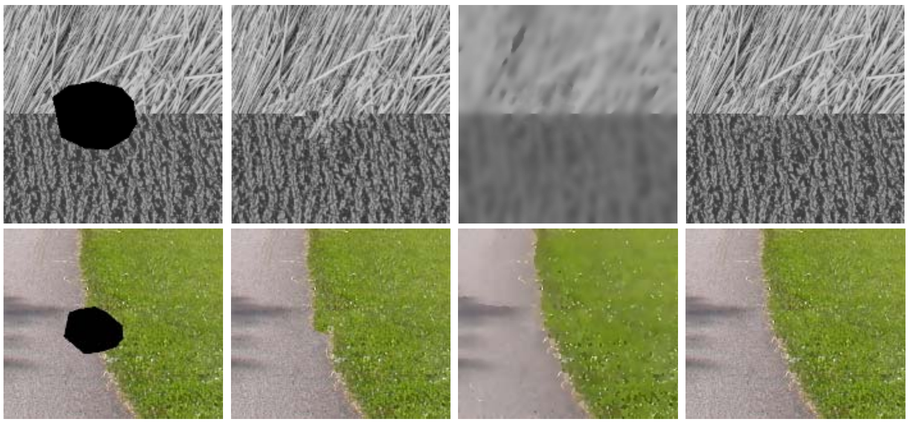


Fig. 2. From left to right: original image with hole, completion using the algorithm of Criminisi et al., completion of I_0, and the final SCIC completion

Table 1. The quality of the completions of our SCIC algorithm are compared with those obtained using the algorithm of Criminisi et al., and also with the modified version of this algorithm, on the image of Fig. 3, with the unknown part of the image centered at fifty equally spaced locations along the diagonal. For the SCIC algorithm we test both Perona-Malik (edge-preserving) smoothing and isotropic Gaussian filtering.

Q	SCIC (Perona-Malik)	SCIC (Gaussian)	Modified Criminisi	Criminisi
1	10%	28%	48%	56%
2	44%	40%	40%	36%
3	46%	32%	12%	8%

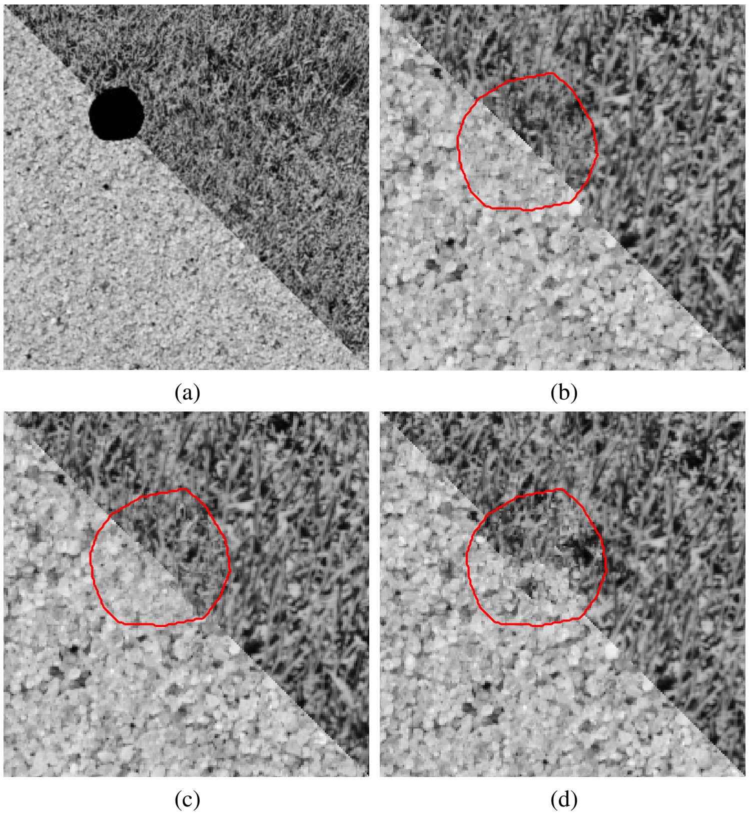


Fig. 3. Assessing quality: (a) Image with hole. (b) A high-quality completion ($Q = 1$). (c) Medium quality ($Q = 2$): the edge is not completely straight. (d) Low quality ($Q = 3$): the edge was not completed correctly. Panels b, c and d are close-ups of the completed region, with the closed curve marking the boundary of Ω_m.

of the SCIC approach for two images: a synthetic image comprised of two textures, and a natural color image with a curved border between the textures.

Next, we perform a systematic comparison for a set of experiments on a synthetic image containing two different textures separated by a diagonal border. The unknown

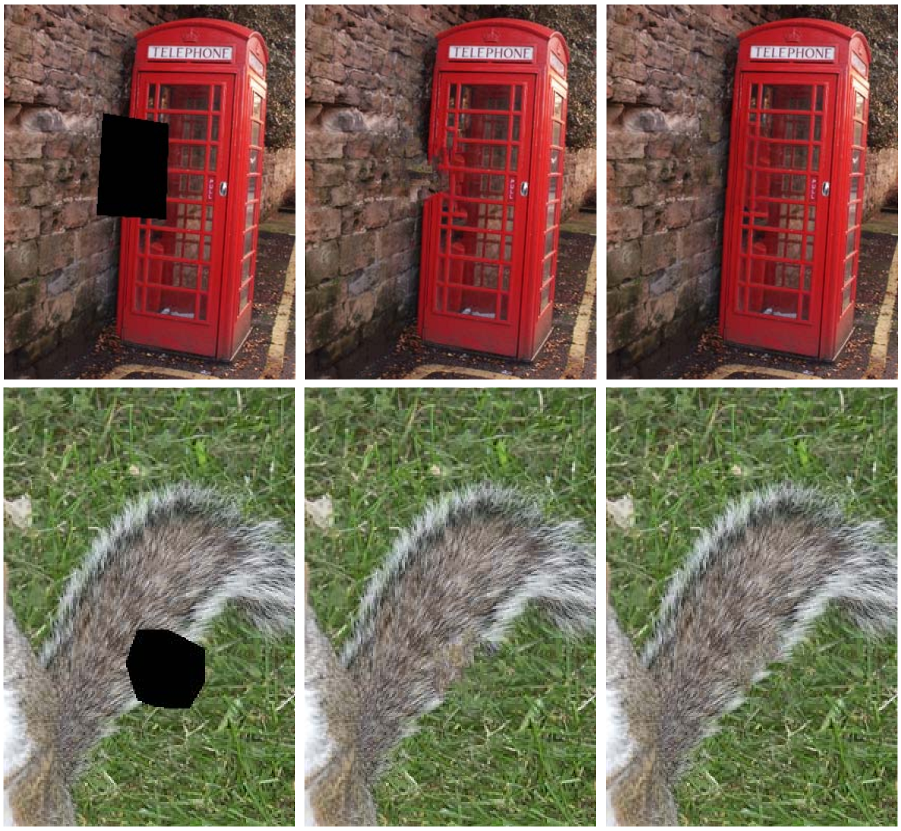


Fig. 4. Left: original image with hole. Middle: result using Criminisi's algorithm. Right: SCIC final result.

region is centered on the border; see example in Fig. 3a. We compare four algorithms for 50 equally spaced locations of the hole along the border. For the SCIC algorithm we used 8 levels of detail generated by the Perona-Malik flow [26]. Then, we repeat the experiment smoothing instead by a convolution with a Gaussian (again, with 8 levels of detail), to test the importance of edge-preserving smoothing. We employ the modified Criminisi algorithm with three allowable patch sizes, both in these two SCIC implementations and as a stand-alone completion method. Finally, we also apply the original algorithm of Criminisi et al. for the same 50 tests. A trinary quality measure, Q, is defined, with $Q = 3$ corresponding to a high-quality completion (essentially perfect), $Q = 2$ for a completion that is slightly flawed but still good, and $Q = 1$ for completions with visible defects. Figs. 2b, 2c, and 2d show typical examples of each of these cases.

The results of the 50×4 experiments are summarized in table 1. The SCIC algorithm scores a 1 only in 10% of the experiments, and a perfect 3 in 46% of the experiments. The average score for SCIC is 2.36. All the failures happened already at the smoothest level, suggesting a potential advantage in also letting the detailed levels guide the smooth-level completions (see discussion in Section 2). The original algorithm of

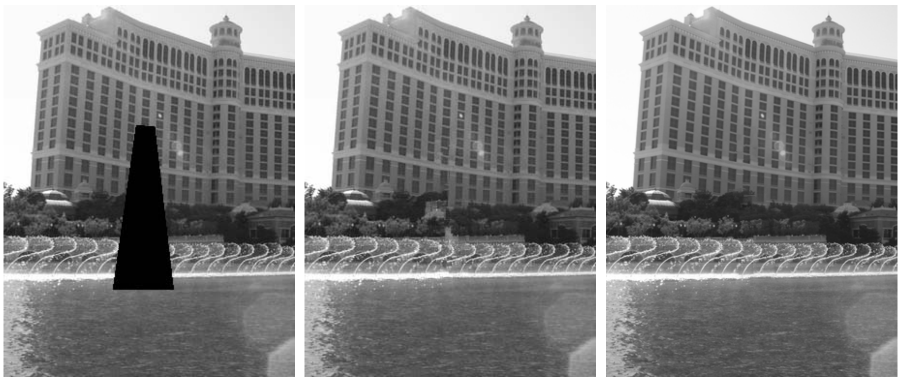


Fig. 5. Left: original image with hole. Middle: result using Criminisi's algorithm. Right: SCIC final result.

Criminisi et al., in contrast, scores a 1 in 56% of the experiments, and a 3 in only 8% of the experiments, with an average score of 1.52. We thus find that SCIC exhibits a dramatic improvement in robustness for this set of experiments. The modified Criminisi algorithm receives an average score of 1.64—a moderate improvement over the basic approach. This demonstrates the crucial role of scale consistency in boosting the performance of a given completion algorithm. Finally, when the Perona-Malik smoother is replaced by a simple isotropic Gaussian filter, the average score is 2.04—a substantial degradation in performance, though still significantly better than the single-level results.

We next test natural images, obtained from [28]. The SCIC algorithm employs 6 levels of detail, with 5 allowable patch sizes $(10^2, 15^2, 20^2, 30^2, 40^2)$ at the smoothest level and 3 allowable patch sizes for the rest of the levels $(10^2, 20^2, 40^2)$. Here and in the previous examples, the patch size for the Criminisi algorithm is 10^2, which is the most common patch size in the modified algorithm.

In Fig. 4 we show two examples with color images. SCIC performs far better than the single-scale approach, which leaves a "bite" in the phone booth and the squirrel's tail. In Figure 5 we show a difficult image containing many different patterns at various scales and a large hole. Again we see that, by using our scale consistent algorithm, all patterns and the borders between them are well reconstructed. The algorithm of Criminisi et al. yields a completion that is flawed in many locations, leaving interrupted water curves in the fountain, mixing patterns, and introducing a part of the building where there should only be trees.

5 Conclusions and Discussion

A new approach for image completion is developed, which uses multiple levels of detail of the image. The method is based on a novel concept of scale-consistency, which requires that the processes of smoothing and completing of images should approximately commute. This framework introduces a new dimension into the patch-based image completion formalism, namely, scale.

These ideas are implemented, employing an image completion algorithm based on [24], and the edge-preserving smoothing algorithms of Perona-Malik for grey-level images and Beltrami flow for color images. Results of experiments—both systematic tests with synthetic images and experiments with natural images—demonstrate a very substantial improvement in the robustness of the completions.

An important issue that requires further investigation is the lack of direct influence of fine-detail images in the hierarchy on the smooth-image completions. This means that failure to complete the least detailed image well is likely to result in overall failure.

Another issue that has not yet been investigated thoroughly is how many levels of detail need to be used, and, more generally, the computational complexity of the algorithm. In light of the expected weak influence of the number of levels on the computational complexity (see subsection 3.4), it is probably best to choose a relatively large number of levels, as we do in our tests. Although the issue of efficient implementation has not yet been thoroughly researched, we expect that the total computational effort spent in the completion stage will be just a fraction more than the cost of completing a single image using the standard basic completion method. To this we need to add the cost of constructing images at varying degrees of detail, which depends on the method used but is usually much smaller than the cost of completing a single image.

References

1. Ballester, C., Bertalmio, M., Caselles, V., Sapiro, G., Verdera, J.: Texture mixing and texture movie synthesis using statistical learning. IEEE Trans. Image Processing **10** (2001) 1200–1211
2. Bertalmio, M., Sapiro, G., Caselles, V., Ballester, C.: Image inpainting. In: SIGGRAPH 2000, Computer Graphics Proceedings, ACM SIGGRAPH (2000) 417–424
3. Chan, T., Shen, J.: Non-texture inpaintings by curvature-driven diffusions. J. Visual Communication and Image Representation **12(4)** (2001) 436–449
4. Telea, A.: An image inpainting technique based on the fast marching method. Journal of Graphics Tools: JGT **9** (2004) 23–34
5. Oliveira, M.M., Bowen, B., McKenna, R., Chang, Y.S.: Fast digital image inpainting. In: Proceeding of International Conference on VIIP. (2001) 261–266
6. Elad, M., Starck, J.L., Querre, P., , Donoho, D.: Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA). Journal on Applied and Computational Harmonic Analysis **19** (2005) 340–358
7. Rares, A., Reinders, M., Biemond, J.: Edge-based image restoration. IEEE Trans. Image Processing **14** (2005) 1454–1468
8. Shih, T., Lu, L.C., Wang, Y.H., Chang, R.C.: Multi-resolution image inpainting. In: Proceedings of the 2003 International Conference on Multimedia and Expo. Volume 1. (2003) 485–8
9. Bonet, J.S.D.: Multiresolution sampling procedure for analysis and synthesis of texture images. In: Computer Graphics Proceedings, ACM SIGGRAPH (1997) 361–368
10. Efros, A., Leung, T.: Texture synthesis by non-parametric sampling. In: ICCV proceedings, Corfu, Greece (1999) 1033–1038
11. Harrison, P.: A non-hierarchical procedure for re-synthesis of complex textures. In: WSCG. (2001) 190–197
12. Wei, L., Levoy, M.: Fast texture synthesis using tree-structured vector quantization. In: SIGGRAPH 2000, Computer Graphics Proceedings, ACM SIGGRAPH (2000) 479–488

13. Hertzmann, A., Jacobs, C.E., Oliver, N., Curless, B., Salesin, D.H.: Image analogies. In: SIGGRAPH 2001, Computer Graphics Proceedings, ACM SIGGRAPH (2001) 327–340
14. Ashikhmin, M.: Synthesizing natural textures. In: Proceedings of the 2001 symposium on Interactive 3D graphics, New York, NY, USA, ACM Press (2001) 217–226
15. Efros, A., Freeman, W.: Image quilting for texture synthesis and transfer. In: SIGGRAPH 2001, Computer Graphics Proceedings, ACM SIGGRAPH (2001) 341–346
16. Kwatra, V., Schödl, A., Essa, I., Turk, G., Bobick, A.: Graphcut textures: Image and video synthesis using graph cuts. ACM Trans. on Graph., SIGGRAPH 2003 **22** (2003) 277–286
17. Zhang, Y., Xiao, J., Shah, M.: Region completion in a single image. EUROGRAPHICS (2004)
18. Jia, J., Tang, C.: Inference of segmented color and texture description by tensor voting. IEEE Trans. Pattern Anal. Mach. Intell. **26** (2004) 771–786
19. Bertalmio, M., Vese, L., Sapiro, G., Osher, S.: Simultaneous structure and texture image inpainting. In Proc. of Computer Vision and Pattern Recognition **2** (2003) 707 –712
20. Drori, I., Cohen-Or, D., Yeshurun, H.: Fragment-based image completion. ACM Trans. Graph. **22** (2003) 303–312
21. Cant, R., Langensiepen, C.: A multiscale method for automated inpainting. In: Proceeding of ESM2003. (2003) 148–153
22. Wexler, Y., Shechtman, E., Irani, M.: Space-time video completion. In Proc. of Computer Vision and Pattern Recognition **1** (2004) 120–127
23. Sun, J., Yuan, L., Jia, J., Shum, H.Y.: Image completion with structure propagation. ACM Trans. Graph. **24** (2005) 861–868
24. Criminisi, A., Perez, P., Toyama, K.: Region filling and object removal by exemplar-based inpainting. IEEE Trans. on Image Processing **13** (2004) 1200–1212
25. Galun, M., Sharon, E., Basri, R., Brandt, A.: Texture segmentation by multiscale aggregation of filter responses and shape elements. In: ICCV. (2003) 716–723
26. Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. IEEE Trans. Pattern Anal. Mach. Intell. **12** (1990) 629–639
27. Spira, A., Kimmel, R., Sochen, N.: Efficient beltrami flow using a short time kernel. In: Proc. of Scale Space 2003, Lecture Notes in Computer Science (vol. 2695), Isle of Skye, Scotland, UK (2003) 511–522
28. Schaefer, G., Stich, M.: UCID - an uncompressed colour image database. Proc. SPIE, Storage and Retrieval Methods and Applications for Multimedia 2004 (2004) 472–480

EXDRAP: An Extended Dead Reckoning Architectural Pattern for the Development of Web-Based DVE Applications

Nerssi Nasiri Amini and Mostafa Haghjoo

Computer Engineering Department, Iran University of Science and Technology
16846-13114, Tehran, Iran
Nasiri_amini@comp.iust.ac.ir, haghjoom@iust.ac.ir

Abstract. Prosperity of distributed 3D applications on the Web heavily depends on the portability and reusability of the content created. Currently, Web3d formats often fall short in resolving such issues. This paper introduces EXDRAP as a hybrid publishing paradigm for declaratively creating Web-based collaborative virtual reality applications which we believe improves portability and reusability. The major issues concerning the development of Web-based CVEs are closely investigated; and an extended *dead reckoning* technique and an optimizing translation mechanism are proposed which reduce the latency (lag) and the amount of memory taken by the browser, respectively. Based on X3D (the successor to VRML) as the ISO standard for real-time computer graphics on the Web, the concepts are successfully implemented and integrated into Jakarta Struts Framework. In order to gain maximum portability, the integration of the X3D browser and the server-side technology is made possible through the use of ECMAScript instead of java on the client end.

Keywords: Extensible 3D (X3D), VRML, Distributed Behavior, Distributed Virtual Environment (DVE), Collaborative Virtual Environments (CVE), ECMAScript, Java, JSP.

1 Introduction

More than a decade has passed since the introduction of declarative 3D virtual reality languages such as VRML to the World Wide Web, yet only few success stories can be told. While large scale legacy systems were successfully transferred to the Web forming enterprises, the newly born VR languages have failed to bring the same media rich and highly interactive content from desktop applications to the Web environment. As for the VRML language, part of the blame can be laid on SGI restructuring and lack of needed support in the late 90s, but the main reason is the language itself. VRML is a declarative language but unlike HTML pages, 3D content creation requires a lot of effort even with the most efficient authoring tools at hand. Declarative languages hide the complexity and let authors create the content with the least programming skills but the side effect is the reduction of reusability and extensibility of the content created. As for the HTML pages, it's not really a big issue to create them from scratch (and if so it's usually resolved with a simple *copy and paste*) but for the 3D content, it's indispensable.

Although sophisticated solutions exist for producing and especially reusing dynamic web pages and other media contents, projects with interactive 3D graphics including DVEs are often developed from scratch. Probably the most prosperous approaches among these (and the one applied in this paper) are those who extend the declarative language by defining some higher level nodes and transforming them into the target language encoding at runtime before delivery [6], [7], [12], [13]. Such approaches let content authors add higher level functionalities to the language and resolve some virtual reality related issues by employing these encapsulated pieces, and probably extending or reusing them later on. The problem with most existing solutions is that they usually tackle with just one proprietary issue and usually don't define a pervasive paradigm covering others.

Another issue taken into consideration in this paper is the integration of VRML97/X3D browser with the server-side technologies. The integration of X3D scene and external applications is made possible through SAI (Scene Authoring Interface) [25]. According to X3D Specification, this interface is to be supported by the conforming browsers using Java or ECMAScript (an internal scripting language) bindings. Java is a wonderful cross platform technology, but the portability seems to end when it is used with VRML97/X3D browsers. Not all X3D browsers support Java since Java support is not required for a conforming implementation of SAI; and if so, they might support different java virtual machines. A conforming browser should merely support ECMAScript. Additionally unlike java (which has to be compiled first, then delivered to the browser), ECMAScript shall be created dynamically at runtime which brings a great deal of flexibility to the publishing paradigms applied.

Many Collaborative Virtual Environments (CVE) have been developed using VRML and External Authoring Interface (EAI) to connect the VRML browser to a java applet which can communicate with a server [9], [11], [20]. Others do this via extensions to the VRML language referred to as SDKs [2], [14]; but few has used JavaScript to support such integration [17] which was first suggested by Cowie [4] and none has implemented DVEs in X3D format using the new ECMAScript binding [27] for supporting distributed behavior.

This paper is organized as follows: The next section relates our work to existing approaches. Section 3 analyzes the applicable paradigms for publishing X3D content on the Web, proposes EXDRAP (EXtended Dead Reckoning Architectural Pattern) model which applies those paradigms and introduces an extended dead reckoning technique on top of the proposed model which would facilitate the development of Web-based DVE applications. Section 4 gives details on the application example which is a multi-player football game. The paper is finished with a discussion on evaluating our work and an outline of future work.

2 Related Work

The four levels of behavior, first coined by Roehl, have been frequently referenced in literature where: direct modification of an entity's attributes defines level 0; the change of an entity's attributes over time constitutes level 1; level 2 comprises a series of calls to level 1 behaviors to perform some task; level 3, after all, is characterized as top-level decision-making [18]. According to Roehl, the distribution of higher levels of behavior over the network results in more limited amount of communication among hosts, and hence less network traffic; but since level 2 and level 3 behaviors are non-deterministic, it's not realistic to use them for behavior distribution. Roehl recommends level 1 for such behavior distribution, but does not suggest how to implement it in declarative virtual reality languages.

Behavior3D [6], an XML-based framework for 3D Graphic Behavior, which is part of the research project CONTIGRA [5], utilizes the level 2 of Roehl's behavior model in order to extend the existing component oriented architecture and additionally support behavior encapsulation. Although Behavior3D brings reusability and extensibility by defining higher level XML-based behavior nodes for deterministic behaviors (such as different states of a virtual laptop or an answering machine), it falls short in providing routines on how to distribute either deterministic or non-deterministic behaviors over the network. However, the idea of content encapsulation into higher level nodes and translating them at runtime has influenced this work.

The first successful implementations of DVE resolving the primary issues associated with distributing behaviors over the network emerged in the Distributed Interactive Simulation (DIS) systems [8]. DIS resolves these issues by introducing a variety of techniques such as the definition of a standard message format, dead reckoning, multicasting and virtual zones (also known as cells) [10].

For more than a decade DIS techniques are being applied to DVE applications to resolve the associated issues. In a DIS project for instance, Macendonia exploited multicasting groups and hexagonal cells to solve the problem of scaling very large distributed simulations [10] and recently Marvie has proposed an extension to VRML97/X3D for massive scenery management in virtual environments [12] which describes cell-to-cell, cell-to-object as well as hybrid visibility relationships using a generic cell representation of static entities in virtual environments and makes it possible to represent both massive indoor and outdoor sceneries.

The DIS Component of X3D is already part of the X3D specification but currently few VRML97/X3D browsers care to support it. The DIS component consists of four X3D nodes: EspduTransform, ReceiverPdu, SignalPdu, and TransmitterPdu. Altogether, these nodes provide the means to send and receive DIS-compliant messages, called Protocol Data Units (PDUs), across the network [25]. The DIS-XML workgroup is currently working on developing DIS support in X3D. The goal is to explore and demonstrate the viability of DIS-XML networking support for X3D to open up the modeling and simulation market to open-standard Web-based technologies. The DIS in X3D is implemented via Java and XML [26].

Roehl implies that deterministic behaviors shall be distributed on higher levels and recommends state machines for doing the job. Viewpoint [22] is a proprietary XML-based format for describing scenes containing so called *Scene Interactors* for defining behaviors. The state machine model used in Viewpoint is suitable for the distribution of deterministic behaviors in DVE applications.

Polys has classified the generation process of Web-based VR content with XML and X3D into four publishing paradigms [16]: Identity Paradigm directly visualizes the static file format; Composition Paradigm permits the composition and delivery of documents "on the fly" in response to the user request; Pipeline Paradigm stores information in an XML-based format and transforms it into the target document on delivery, using XML technologies such as XSLT [23]; Hybrid Paradigm combines Pipeline and Composition Paradigms to gain the maximum flexibility. The publishing paradigm introduced in this paper, relies on the Hybrid Paradigm classified by Polys.

Bitmanagement company, the producer of BS Contact VRML97/X3D browser and other 3D related authoring tools, has proposed some authoring approaches as *Tips and Tricks* to optimize the rendering process of the 3D scenes in their browser [3]. Among these optimizing approaches, some are also applicable to other browsers, and some shall be applied automatically to the 3D content before delivery. The utilization of such optimizing translation techniques have been taken into consideration in EXDRAP, and are implemented in the application example.

3 X3D and Web-Based DVE Applications

The X3D ISO standard defines an XML-based, royalty free file format and a runtime engine for real time 3D content and applications running on a network and shall be integrated with other technologies to support distributed interactive virtual environments and thus is considered a suitable basis for this work.

In order to maintain such integration, first in the following section, possible publishing paradigms for Web-based DVE content generation are analyzed. In section 3.2, EXDARP, the proposed architectural pattern applied in our work, has been discussed. Next we have introduced an extension to dead reckoning applying that pattern.

3.1 Applicable Publishing Paradigms

In order to publish DVE content on the Web, the paradigm applied must maintain maximum flexibility by supporting potential extensions to X3D. It must also make it possible to integrate the application with other emerging technologies on the web. Probably the most routine approach toward VRML based dynamic content publishing on the Web has been to populate static templates with the dynamic contents which are usually fetched from data sources [4], [17], [24]. This approach, known as composition paradigm, is supported by many well-established and mature server-side technologies mainly referred to by server-side scripting languages.

With X3D (as an XML-based language), the powerful XML related tools and the available APIs, alternative approaches shall be additionally taken. These include the use of XSLT to transform higher level nodes (extensions to the language) into the target language [6], [15] or the employment of XML parsers to traverse the document and optimize the result by changing the document structure. All these XML related approaches are known as the pipeline paradigm.

Composition and pipeline paradigms each facilitate different needs and if applied jointly can bring highest flexibility to DVE applications. In our work, we have proposed a hybrid model which uses these two paradigms as separate layers which are discussed in the following section.

3.2 The Proposed Architectural Pattern

EXDRAP (figure 1) applies a model-view-controller (MVC) design pattern in order to separate the interface from control logic. The decision maker employs the control logic of the DVE application and is made up of two components: controller and dispatcher. Controller component provides a centralized entry point for client requests which prevents duplicate code, commingled view content and coalescent view navigation. Controller applies the domain specific business rules and also stores the persistent VR and business related data. Dispatcher component is responsible for the synchronization between hosts and collision detection. The dispatcher, in conjunction with the controller, resolves race conditions before forwarding the request to an appropriate web publisher.

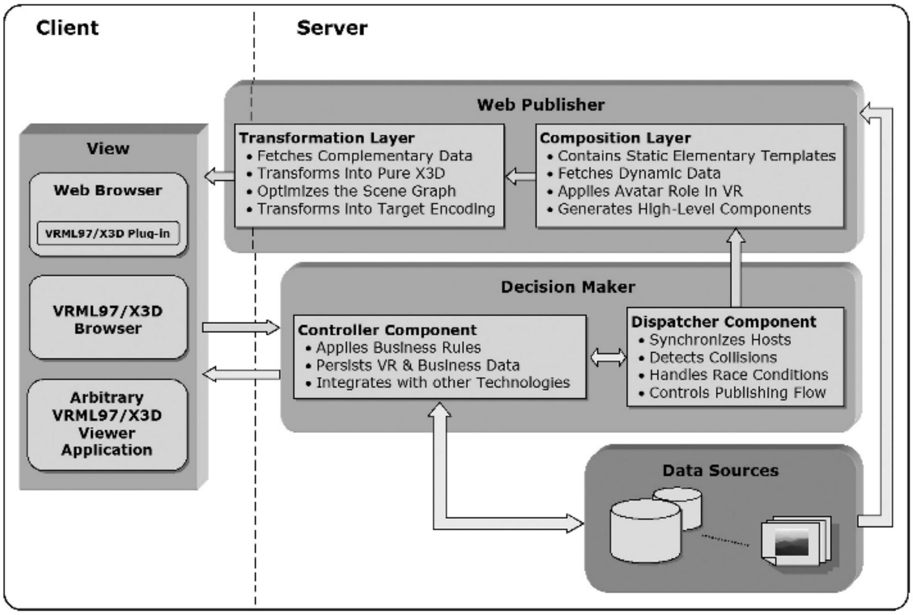

Client
Server
Web Publisher
Transformation Layer
• Fetches Complementary Data
• Transforms Into Pure X3D
• Optimizes the Scene Graph
• Transforms Into Target Encoding
Composition Layer
• Contains Static Elementary Templates
• Fetches Dynamic Data
• Applies Avatar Role In VR
• Generates High-Level Components
View
Web Browser
VRML97/X3D Plug-in
VRML97/X3D Browser
Arbitrary VRML97/X3D Viewer Application
Decision Maker
Controller Component
• Applies Business Rules
• Persists VR & Business Data
• Integrates with other Technologies
Dispatcher Component
• Synchronizes Hosts
• Detects Collisions
• Handles Race Conditions
• Controls Publishing Flow
Data Sources

Fig. 1. The architectural pattern proposed in EXDRAP

Web publisher plays the view part as a double layer component. The first layer which applies a composition paradigm contains the elementary static templates which are processed and populated at runtime with dynamic data (fetched from available data sources) employing various scripting languages. This layer is also responsible for employing the avatar's role in the DVE application.

In addition to X3D nodes, the output of the composition layer shall be an interleaved mixture of any arbitrary higher level XML-based nodes (extensions to X3D). In the transformation layer, the XML related technologies, tools and APIs are utilized to translate this raw output into a pure VRML97/X3D encoding which is supported by the client. Additionally some extra language optimizers shall be applied on the result to exceed the performance of the client's browser. As it can be seen in the figure, transformation layer is positioned between the server and client meaning that part of the transformation process might be performed by the viewer on the client end.

While the web publisher component utilizes diverse techniques using a hybrid paradigm to form a basis for a more flexible content generation, in order to maintain highest portability, the integration of client browser with the server is kept simple and is merely done through stateless http requests.

3.3 Extended Dead Reckoning

DIS dead reckoning technique is a magnificent approach toward the distribution of behaviors in networked virtual environments. Unfortunately this approach works particularly well only for the problem domain it's designed for: military simulations. Roehl suggests generalizing the technique by supporting the distribution of other kinds of his so called level 1 behaviors.

In VRML97/X3D, level 1 behaviors shall be produced with the use of arbitrary combinations of Routes, Interpolator nodes and Event utilities. There are two alternatives for initiating such behaviors: the Sensor nodes and the Script node. Sensor nodes are embedded event generators of the X3D language which can initiate behaviors by triggering events based on user interactions, environmental collisions or the passing of time. Script node in contrast is the only access point for programmatic scene manipulation. This programmatic interface is provided through the Scene Access Interface (SAI) internally from Script nodes (the one applied in our work) or externally from other application programs (called scripting environments). In addition to programmatic scene manipulation, Script node makes it possible to integrate the X3D scene with server-side technologies.

We have generalized the dead reckoning technique in X3D by assigning for each virtual object in the scene (for instance a vehicle or a ball) a script node which facilitates the manipulation access to the object's different dynamic entity attributes and maintains a protocol for sending and receiving update and keep-alive messages to the server. In the proposed model discussed in section 3.2, such script nodes are generated dynamically in the composition layer according to client's avatar role in the virtual environment. In our implementation, explained in detail in section 4, we have defined a set of tag libraries which accept an entity's DEF id, its target fields and their type as input attributes and encapsulate the whole generation process of the Script node (its fields and the ECMAScript code to connect the server) and the relevant Routes, Interpolators, Triggers, Sequencers and Sensors.

The event flow to support extended dead reckoning in EXDRAP is as follows: Any event in the scene, defined as shared behaviors, is routed to pre-generated Scripts nodes which results in a message update call to the server as an http request. The controller component on the server (if no collision or race condition is detected) will add this event and its parameters to a singleton collection for other DVE participants to refer to and update their scene accordingly. Since the http request is a one way connection (shall not be initiated by the server), each client has to periodically refer to server to update their scene. The aspect of each client having to periodically contact the server, even when there's no update message available, is not really a great drawback in our approach; since these periodic contacts can act as keep-alives which are really crucial to the dead reckoning technique.

While update messages of non-shared behaviors shall be reported to server asynchronously (send a report to the server and in the mean time apply the change to the scene without waiting for server confirmation), in order to handle race conditions on shared behaviors and prevent the world instances to get out of sync, the update messages have to be reported in synchronous manner. This means that before applying any change to the scene in result of user interaction with a shared behavior (kicking a ball), client must make sure its update message is accepted (no other user has already kicked it in the mean time) by the server and the event is added to the singleton collection. This means that the client has to always wait for the server confirmation. However, as Presser suggests this would be a trade off between synchronization and efficient communication [17].

4 Application Example: A Football Game

Our implemented application which applies the introduced concepts and extends dead reckoning is a multi-player football game (figure 2). Four roles were defined for the participating avatars: the spectators who can not enter the football field nor interact with any entity in the scene; the players who are just able to kick the ball; the goalkeeper in addition to kicking can also hold the ball and drop it in the penalty area; the referee who can hold the ball at any point of the football field.

The application was developed under Apache Struts framework [1]. Apache Struts is an open-source framework which encourages developers to adopt an MVC architecture for developing J2EE web applications. Struts is a very well documented, mature and popular framework for building front ends to Web-based Java applications and that's why although there are newer so called *light weight* MVC frameworks such as Spring [19] and Tapestry [21] available, Struts was chosen for our implementation.

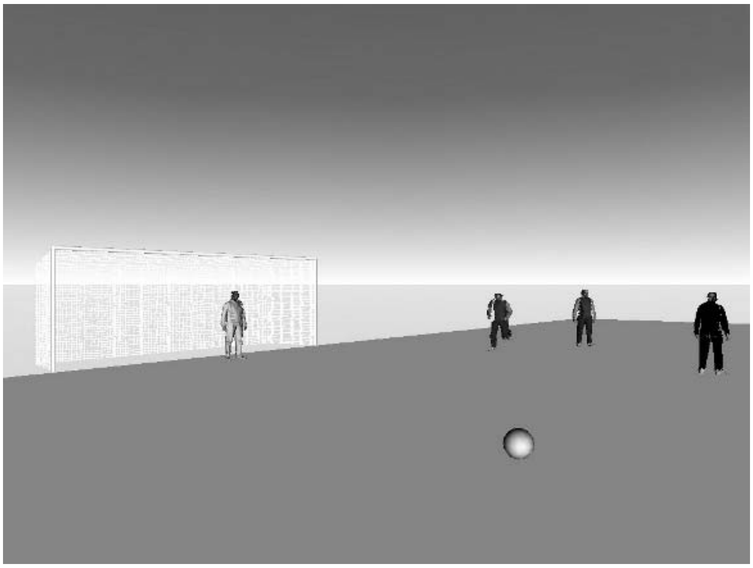


Fig. 2. A screen shot of the implemented application using EXDRAP

The controller and the dispatcher are implemented by ActionForms and DispatchAction forming the decision maker component of the proposed framework. Update messages populate a singleton collection which is shared between participants of the virtual environment. With each update message the current timestamp and the message initiator's session id are assigned so that if no keep-alive arrives to update that timestamp the update message will be automatically canceled after a while (A Thread checks this periodically). Otherwise other hosts will update their scenes according to the update messages provided in the singleton collection and add their session id to that message (so that they won't receive it next time). Another thread is also employed to remove the update messages that are already received by all the participating hosts from the collection and store them in database. This will be useful for the playback of the virtual environment.

The JSP pages are perfect for the implementation of the composition layer but in order to implement the second layer of the web publisher component, we had to add a request filter to the framework and chain it to an object which extends the HttpServletResponse and overrides its getWriter() method. By doing this we managed to acquire the output result of the JSP pages and employ some additional transformational processing before delivering the content to the client.

As mentioned earlier JSP pages generate the needed elements to implement the extended dead reckoning, which is mainly done via a set of pre-defined tag libraries:

```
<Transform DEF="ball">
  <Inline url="http://localhost:8988/dve/ball.x3d" />
</Transform>
<dve:dr objGroup="ballBehavior" sharedBehavior="true" >
  <dve:attribute defId="ball" field="translation"
    fieldType="SFVec3f" behaviorType="straight" />
  <dve:attribute defId="ball" field="rotation"
    fieldType="SFRotation"  behaviorType="iterative" />
</dve:dr >
```

The little piece of code above shows how the distributed behavior of the ball is defined in the JSPs. At runtime, the dve tag is replaced by the Routes, Interpolators, Triggers, Sequencers and Sensors and a Script node needed to distribute the desired behavior over the network. The ECMAScript code generated by the dve tag and used by the Script node would be as follows:

```
ecmascript:
function initialize(){}
function keepAlive(){
  reqUrl=new
MFString("http://localhost:8988/dve/KeepAlive.do");
Browser.createVrmlFromURL(reqUrl,ballBehavior,applyAvai
lableUpdates);

}

function ballTranslation(){

  reqUrl=new
MFString("http://localhost:8988/dve/UpdateScene.do?meth
od=addTranslation&id=ball");

  Browser.createVrmlFromURL(reqUrl, ballBehavior,
applyAvailableUpdates);

}

function ballRotation(){

  reqUrl=new
MFString("http://localhost:8988/dve/UpdateScene.do?meth
od=addRotation&id=ball");

  Browser.createVrmlFromURL(reqUrl, ballBehavior,
applyAvailableUpdates);

}

function applyAvailableUpdates(val,ts){

  if(val.length!=0){

    //applies changes to the scene according to the
contents recieved

  }

}
```

In the translation layer we have used one of the techniques suggested by Bitmanagement to optimize the result scene a bit for the browser to render. The idea is to use DEF/USE as much as possible to reduce delivery content size and run-time memory usage especially for Appearance Material ImageTexture, MovieTexture and etc. In our implementation we have used JDOM to search the X3D structure and remove the url field of the redundant textures and replace them with USE attribute referencing the one which was first appeared in the document.

Since most X3D browsers apply the same policy on the downloading order of the external content (first encountered - first loaded) one can even dictate the downloading order of the textures by transferring them all to the top of the document, applying the ordering and referencing them through the rest of the document. But since the usefulness of such ordering highly depends on the semantics of the scene, defining such policies is not considered in our work.

```
Group {
  children [
    DEF A1 Appearance { } DEF A2 Appearance { ...}
    ....
  ]
}
Shape { appearance USE A1 }
```

Finally before content delivery, based on the encodings supported by the X3D browser, it might be needed to use XSLT to transform the language to the proper encoding. In our implementation, this has been done via Xalan API [28]. Xalan is a popular open source software component from the Apache Software Foundation that implements the XSLT XML transformation language and the XPath XML query language.

5 Discussion and Future Work

In this paper a flexible architectural pattern for the development of Web-based distributed virtual reality applications was introduced that shall be applied to non-distributed applications as well, since the approach itself also enhances the development of non-distributed virtual reality application areas such as arbitrary 3D visualizations. We believe that in addition to simplicity, this approach will maintain the reusability, extensibility and portability of the content created. The composition and transformation layers applied in this framework are flexible enough to support the implementation of VR applications utilizing any existing or future technique which addresses some virtual reality domain related issue.

In this paper we also introduced an extended dead reckoning technique and a scene optimizer mechanism, applying the proposed model and implemented both on Jakarta Struts Framework. A rich set of behavior tags was defined and implemented to support the automatically generation process of arbitrary shared behaviors.

The results have been satisfactory. The amount of the latency present in our application is inevitable and is much due to the nature of the Web environment itself, since even pinging a website with a normal bandwidth link would take about half a second. This latency shall be minimized by reducing the periodic request intervals on the client end. However, very short intervals result in saturated network traffic.

As future work other successful techniques (addressing the main virtual reality domain related challenges) such as virtual zones and multicasting should be taken into consideration and implemented in the framework to guarantee scalability. The singleton collection of message updates shall be divided into partitions each representing a virtual zone in the VR. Then visibility rules shall be applied on the server to decide which update messages shall be received by the participating hosts according to their location in the virtual environment.

Finally with the definition of more complex behaviors and the integration of DVE applications with other server-side technologies (such as EJB or Web Services), it would be possible to offer existing real world Web applications (for instance online shopping and E-learing systems) in an innovative fashion.

References

1. Apache Struts Framework. Available: http://struts.apache.org/ (2006)
2. BLAXXUN. Blaxxun technologies. Available: http://www.blaxxun.com (2005)
3. BS Contact VRML, Tips & Tricks. Available: http://www.bitmanagement.de/developer/?page=/contact/tips.html (2006)
4. COWIE, D.: Use JSP to Create Your Own VRML World. Available: http://builder.com.com/5100-6371-1050067.html (2005)
5. Dachselt, R., Hinz, M., Meißner, K.: CONTIGRA: An XML-Based Architecture for Component-Oriented 3D Applications. In Proceeding of the 7th International Conference on 3D Web Technology (Web3D '02), ACM Press, Tempe, Arizona, USA (2002) 155-163
6. Dachselt, R., Rukzio, E.: BEHAVIOR3D: An XML-Based Framework for 3D Graphics Behavior. In Proceeding of the 8th International Conference on 3D Web Technology (Web3D '03), ACM SIGGRAPH, Saint Malo, France (2003) 101-ff
7. García, P., Montalà, O., Pairot, C., Rallo, R., Skarmeta, A.F.G.: MOVE: Component Groupware Foundations for Collaborative Virtual Environments. Proceedings of the 4th International Conference on Collaborative Virtual Environments (CVE '02), Bonn, Germany (2002) 55 - 62
8. Institute of Electrical and Electronics Engineers, International Standard, ANSI/IEEE Std 1278-1993, Standard for Information Technology, Protocols for Distributed Interactive Simulation (1993)
9. Lovegrove, S., Brodlie, K.: Collaborative Research within a Sustainable Community: Interactive Multi-user VRML and Visualization. In Proceedings of the Eurographics UK Conference (1998) 53–68
10. Macedonia, M.R., Zyda, M.J., Pratt, D.R., Brutzman, D.P., Barham, P.T.: Exploiting Reality with Multicast Groups: A Network Architecture for Large-Scale Virtual Environments. In Proceedings of Virtual Reality Annual International Symposium, Research Triangle Park, NC (1995) 2-10
11. Manoharan, T., Taylor, H., Gardiner, P.: A Collaborative Analysis Tool for Visualization and Interaction with Spatial Data. In Proceedings of the 7th International Conference on 3D Web Technology (Web3D '02), ACM Press, Tempe, Arizona, USA (2002) 75–83
12. Marvie, J.-E., Bouatouch, K.: A VRML97-X3D Extension for Massive Scenery Management in Virtual Worlds. In proceedings of the 9th international conference on 3D Web Technology (Web3D '04), ACM SIGGRAPH, Monterey, Ca., USA (2004) 145-153
13. McIntosh, P., Hamilton, M., van Schyndel, R.: X3D-UML: enabling advanced UML visualisation through X3D. In proceedings of the 10th international conference on 3D Web Technology (Web3D '05), ACM Press, New York, NY, USA (2005) 135-142
14. Picard, S.L.D., Degrande, S., Gransar, C., Chaillou, C., Saugis, G.: VRML Data Sharing in the Spin-3D CVE. In Proceedings of the 7th International Conference on 3D Web Technology (Web3D '02), ACM Press, Tempe, Arizona, USA (2002) 165–172
15. Polys, N.F.: Stylesheet Transformations for Interactive Visualization: Towards a Web3D Chemistry Curricula. In Proceedings of the 8th International Conference on 3D Web Technology (Web3D '03), ACM SIGGRAPH, Saint Malo, France (2003) 85-90
16. Polys, N.F.: Publishing Paradigm with X3D. In Chaomei Chen (Ed.), Information Visualization with SVG and X3D, Springer-Verlag (2005)
17. Presser, C.G.M.: A Java Web Application for Allowing Multi-user Collaboration and Exploration of Existing VRML Worlds, In Proceedings of the 10th International Conference on 3D Web Technology (Web3D '05), School of Informatics, University of Wales, Bangor UK (2005) 85-92

18. Roehl, B.: Some Thoughts on Behavior in VR Systems. Available: http://ece.uwaterloo.ca/~broehl/behav.html (1995)
19. Spring Framework. Available: http://www.springframework.org/ (2006)
20. Swing, E.: Adding Immersion to Collaborative Tools. In Proceedings of the 5th Symposium on Virtual Reality Modeling Language, ACM Press, Monterey, California, USA (2000) 63–68
21. Tapestry Framework. Available: http://jakarta.apache.org/tapestry/ (2006)
22. Viewpoint. Available: http://www.viewpoint.com (2006)
23. White C.: Mastering XSLT. San Francisco: Sybex (2002)
24. Walczak, K., Cellary, W.: Building Database Applications of Virtual Reality with X-VRML. In Proceedings of the 7th International Conference on 3D Web Technology (Web3D '02), ACM Publisher, Tempe, Arizona, USA (2002) 111-120
25. X3D Abstract: ISO/IEC 19775:2004/Am1:2006. Available: http://www.web3d.org/x3d/specifications/ISO-IEC-19775-X3DAbstractSpecification_+_Amendment1_to_Part1/
26. X3D DIS-XML Working Group. Available: http://www.web3d.org/x3d/workgroups/dis/ (2006)
27. X3D language bindings: ECMAScript: ISO/IEC 19777-1:2005. Available: http://www.web3d.org/x3d/specifications/ISO-IEC-19777-1-X3DLanguageBindings-ECMAScript/ (2005)
28. Xalan Component. Available: http://xml.apache.org/xalan-j/ (2006)

Optimal Parameterizations of Bézier Surfaces

Yi-Jun Yang[1,2], Jun-Hai Yong[1], Hui Zhang[1],
Jean-Claude Paul[1], and Jiaguang Sun[1,2]

[1] School of Software, Tsinghua University, Beijing, China
[2] Department of Computer Science and Tech., Tsinghua University, Beijing, China

Abstract. The presentation of Bézier surfaces affects the results of rendering and tessellating applications greatly. To achieve optimal parameterization, we present two reparameterization algorithms using linear Möbius transformations and quadratic transformations, respectively. The quadratic reparameterization algorithm can produce more satisfying results than the Möbius reparameterization algorithm with degree elevation cost. Examples are given to show the performance of our algorithms for rendering and tessellating applications.

1 Introduction

In Computer Aided Geometric Design (CAGD), algorithms for rendering, intersecting and tessellating curves and surfaces are generally based on their parameterization rather than their intrinsic geometry. The quality of the parameterization influences the results of the applications greatly. Uniform speed on parameter lines-i.e., equal increments in the parameter defining equal increments in arc length) is identified as an important character of optimal parameterizations for many applications such as computer numerical control(CNC), texture mapping and tessellating.

In the past 10 years, how to achieve uniform speed on the Bézier curves has been extensively studied in many literatures such as [1,2,3,4,5,6,7,8,9,10,11]. Farouki [1] studied the optimal reparameterization of Bézier curves. In [1], arc-length parameterization is identified as the optimal parameterization of Bézier curves. By minimizing an integral which measures the deviation from arc-length parameterization, the optimal representation is obtained by solving a quadratic equation . Jüttler [2] presented a simplified approach to Farouki's result by using a back substitution in the integral. Costantini [3] obtained closer approximations to arc-length parameterization by applying composite reparameterizations to Bézier curves.

To the author's knowledge, little attention has been paid to the Bézier surface reparameterization. The representation of Bézier surfaces influences the results of rendering and tessellating applications greatly. Given a Bézier surface (see Figure 1 (a)), we map a chessboard texture image onto the surface. However the texture mapping result (see Figure 1 (c)) deviates from our expectation. There is much unwanted variation in the texture mapping result. Many surface tessellating algorithms [12,13,14] first triangulate the parameter domain of the

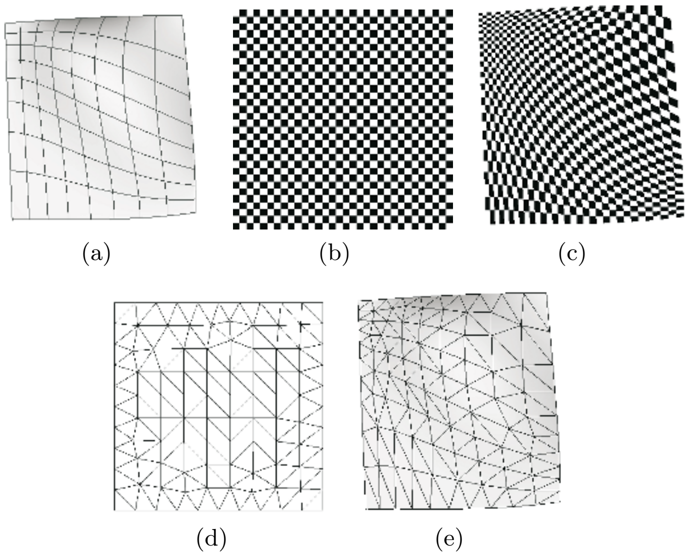


Fig. 1. Texture mapping and tessellating results of a Bézier surface: (a) Bézier surface and its polynomial parameterization; (b) texture image; (c) texture mapping result; (d) triangulation of the parameter domain; (e) tessellating result by mapping triangles in (d) to the surface

given surface (see Figure 1 (d)). Then the 2D triangles are mapped onto the surface to obtain the 3D triangles. Also the representation of Bézier surfaces affects the final results greatly (see Figure 1 (e)). From our point of view, the lack of satisfying representation is the bottleneck for rendering and tessellating algorithms to achieve high quality results.

In this paper, we identify the optimal reparameterizations of Bézier surfaces within the realm of linear and quadratic reparameterizations. Uniform parameter line is an important character of optimal reparameterizations. To achieve uniform speed on parameter lines, linear Möbius transformation is first studied. However the Möbius reparameterizations can not change the shape of the parameter lines. What changes is the distribution of the parameter lines. To obtain more uniform parameter lines, a quadratic reparameterization algorithm is presented. First the Möbius transformations are applied to some definite parameter lines to yield uniform speed on the selected lines. By interpolating the Möbius transformation coefficients using a least square technique, we then get the quadratic reparameterization coefficients. The examples indicate that, in practice, the algorithm produces significantly more uniform parameter lines across the Bézier surfaces. The main contribution of our work can be summarized as follows:

- We show that linear Möbius transformations can not change the shape of the parameter lines. What changes is the distribution of the parameter lines.
- We present a quadratic reparameterization algorithm to obtain more uniform parameter lines across the Bézier surfaces.

The paper is organized as follows. Section 2 describes how to obtain uniform speed on parameter lines using Möbius transformations. Section 3 shows how to use the quadratic reparameterization to achieve more uniform parameter lines for Bézier surfaces. In Section 4, we conclude the paper.

2 Möbius Reparameterization

Given a Bézier surface of the following form

$$\mathbf{X}(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} B_i^m(u)B_j^n(v)\mathbf{P}_{i,j}, \quad u \in [0,1], v \in [0,1], \tag{1}$$

where the $\mathbf{P}_{i,j}$ are the control points, the $B_i^m(u)$ and $B_j^n(v)$ are the Bernstein polynomials, we try to obtain uniform speed for finitely many parameter lines $\mathbf{X}(u_i, v)$, for certain constant values of u_i, and $\mathbf{X}(u, v_j)$, for certain constant values of v_j. For these parameter lines, we take the uniform speed functional used in Farouki's paper [1]. To start with, we just consider the four boundary curves and try to obtain uniform speed there. To obtain uniform speed on the four boundaries, the following integral function

$$J(\alpha,\beta) = \int_0^1 ||\frac{\partial \mathbf{X}(u,0)}{\partial u}||^2 du + \int_0^1 ||\frac{\partial \mathbf{X}(u,1)}{\partial u}||^2 du$$

$$+ \int_0^1 ||\frac{\partial \mathbf{X}(0,v)}{\partial v}||^2 dv + \int_0^1 ||\frac{\partial \mathbf{X}(1,v)}{\partial v}||^2 dv \tag{2}$$

is adopted. The integral measures the deviation of the four boundaries from the uniform-speed paramterizations of the four boundaries. It is convenient to adopt normalized coordinates for the four boundaries such that $J = 4$ if the four boundary curves are of arc-length parameterization, otherwise, $J > 4$ [1]. Each parameter is subjected to a Möbius transformation as follows.

$$u = u(s) = \frac{(\alpha - 1)s}{2\alpha s - s - \alpha}, \tag{3}$$

and

$$v = v(t) = \frac{(\beta - 1)t}{2\beta t - t - \beta}. \tag{4}$$

Applying the transformations (3) and (4) to surface (1) results in the rational Bézier surface

$$\mathbf{X}(s,t) = \frac{\sum\limits_{i=0}^{m}\sum\limits_{j=0}^{n} B_i^m(s)B_j^n(t)\omega_{i,j}\mathbf{P}_{i,j}}{\sum\limits_{i=0}^{m}\sum\limits_{j=0}^{n} B_i^m(s)B_j^n(t)\omega_{i,j}}, \quad s \in [0,1], t \in [0,1],$$

with $\omega_{i,j} = (1-\alpha)^i \alpha^{m-i}(1-\beta)^j \beta^{n-j}$. Here we want to choose the transformations (3) and (4) such that $J(\alpha, \beta)$ becomes as small as possible. With the help of the chain rule, we get from Equation (2)

$$J(\alpha,\beta) = \int_0^1 ||\frac{\partial \mathbf{X}(s,0)}{\partial s}||^2 \frac{(1-\alpha+2s\alpha-s)^2}{\alpha(1-\alpha)} ds + \int_0^1 ||\frac{\partial \mathbf{X}(s,1)}{\partial s}||^2 \frac{(1-\alpha+2s\alpha-s)^2}{\alpha(1-\alpha)} ds$$

$$+ \int_0^1 ||\frac{\partial \mathbf{X}(0,t)}{\partial t}||^2 \frac{(1-\beta+2t\beta-t)^2}{\beta(1-\beta)} dt + \int_0^1 ||\frac{\partial \mathbf{X}(1,t)}{\partial t}||^2 \frac{(1-\beta+2t\beta-t)^2}{\beta(1-\beta)} dt$$

The solution satisfies the following two equations

$$0 = \frac{\partial J(\alpha,\beta)}{\partial \alpha}, \tag{5}$$

and

$$0 = \frac{\partial J(\alpha,\beta)}{\partial \beta}. \tag{6}$$

First, we simplify Equation (5) as follows.

$$0 = \frac{\partial J(\alpha,\beta)}{\partial \alpha} = \int_0^1 ||\frac{\partial \mathbf{X}(s,0)}{\partial s}||^2 \frac{-B_0^2(s)B_0^2(\alpha) + B_2^2(s)B_2^2(\alpha)}{\alpha^2(1-\alpha)^2} ds +$$

$$\int_0^1 ||\frac{\partial \mathbf{X}(s,1)}{\partial s}||^2 \frac{-B_0^2(s)B_0^2(\alpha) + B_2^2(s)B_2^2(\alpha)}{\alpha^2(1-\alpha)^2} ds = \frac{P_1 B_0^2(\alpha) + Q_1 B_2^2(\alpha)}{\alpha^2(1-\alpha)^2}$$

with the coefficients

$$P_1 = \int_0^1 -(||\frac{\partial \mathbf{X}(s,0)}{\partial s}||^2 + ||\frac{\partial \mathbf{X}(s,1)}{\partial s}||^2) B_0^2(s) ds,$$

and

$$Q_1 = \int_0^1 (||\frac{\partial \mathbf{X}(s,0)}{\partial s}||^2 + ||\frac{\partial \mathbf{X}(s,1)}{\partial s}||^2) B_2^2(s).$$

Note that $P_1 < 0 < Q_1$ holds. Hence we get exactly one root of $P_1 B_0^2(\alpha) + Q_1 B_2^2(\alpha) = 0$ with $0 < \alpha < 1$. Equation (6) can be solved using a similar method. Also the polynomial form of the coefficients can be easily obtained. In Figure 2, we give an example to show the improved parameter speed that can be realized by the Möbius reparameterizations. The parameter lines corresponding to a fixed parameter increment in another parameter for the original representation and the optimal rational representation are given in Figure 2(a) and Figure 2(b) respectively. For the surface in Figure 2(a), the original polynomial representation has $J = 5.139$. The optimal reparameterization occurs for

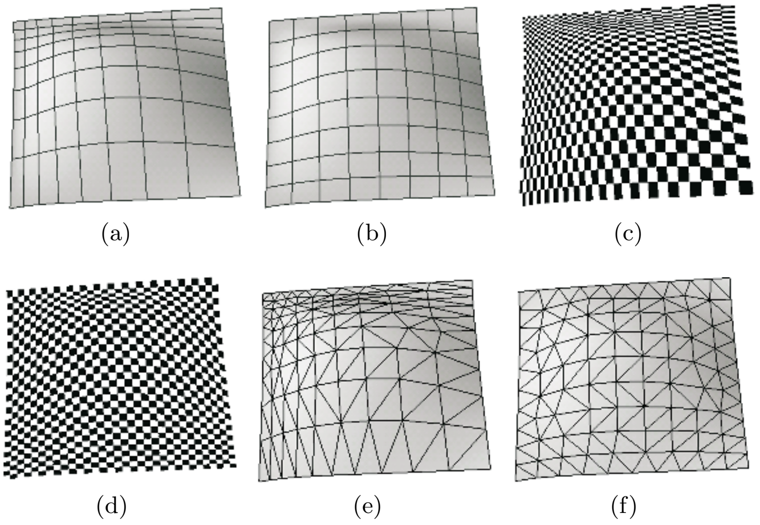


Fig. 2. Texture mapping and tessellating results of a Bézier surface: (a) Bézier surface and its polynomial parameterization; (b) Bézier surface and its optimal parameterization; (c) texture mapping result of the surface; (d) texture mapping result of the reparameterized surface; (e) tessellating result of surface (a) by mapping Figure 1(d) to the surface; (f) tessellating result of surface (b) by mapping Figure 1(d) to the reparameterized surface

$\alpha = 0.274, \beta = 0.311$ - yielding a value, $J = 4.019$ that yields satisfying results for rendering and tessellating applications. For the Bézier surface in Figure 2(a), the original polynomial parameterization has $J = 4.524$. The optimal reparameterization occurs for $\alpha = 0.482, \beta = 0.432$ - yielding a value, $J = 4.469$ that yields a surface similar to the original surface (see Figure 3). From Figures 2 and 3, we can see that the linear Möbius transformations can not change the shape of the parameter lines. What changes is the distribution of the parameter lines. Thus the parameter lines will not become more uniform after linear Möbius reparameterizations for some surface cases. Also the Möbius reparameterizations are not expected to yield dramatic improvements in the parameter speed for arbitrary Bézier surfaces. The optimal reparameterization coefficients (using the integral in [1]) for the four boundary curves are shown in Table 1. We can see that if the changes of the parameter speed across the two opposite boundaries are not consistent (the optimal reparameterization coefficient for one boundary is greater than 0.5 while the reparameterization coefficient for another opposite boundary is less than 0.5), the Möbius transformations take little effect.

Table 1. Optimal reparameterization coefficients for Bézier curves

Curve	$S(u,0)$	$S(u,1)$	$S(0,v)$	$S(1,v)$
surface in Figure 2(a)	0.268	0.279	0.294	0.324
surface in Figure 1(a)	0.290	0.676	0.435	0.428

3 Quadratic Reparameterization of Bézier Surfaces

Given a Bézier surface, each parameter is subjected to a transformation as follows.

$$u = u(s) = \frac{(\alpha - 1)s}{2\alpha s - s - \alpha}, \tag{7}$$

and

$$v = v(t) = \frac{(\beta - 1)t}{2\beta t - t - \beta}, \tag{8}$$

where

$$\alpha = \alpha_1 t + \alpha_2(1-t) \text{ and } \beta = \beta_1 s + \beta_2(1-s).$$

Here in order to achieve more uniform parameter lines, we need additional free parameters. We choose α in (7) as a linear function of t, with coefficients α_1 and α_2, and β in (8) as a linear function of s, with coefficients β_1 and β_2. This will lead to an optimization problem with 4 variables, and it will also raise the degree of the surface accordingly.

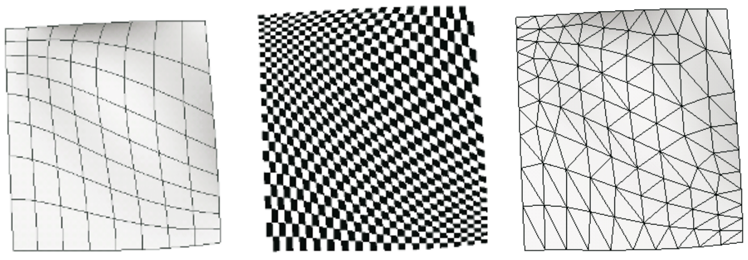

(a) (b) (c)

Fig. 3. Texture mapping and tessellating results of a Bézier surface: (a) Bézier surface and its optimal parameterization; (b) texture mapping result of reparameterized surface (a); (d) tessellating result by mapping Figure 1(d) to the reparameterized surface

3.1 Computing the Control Points and Weights of the Reparameterized Surface

The new surface would be of degree $(m+n) \times (m+n)$. The new Bézier functions are computed as follows.

$$B_i^m(s) = \frac{B_i^m(u)(\alpha_2 v - \alpha_1 v - \alpha_2)^{m-i}(\alpha_1 v + \alpha_2 - \alpha_2 v - 1)^i}{(2u\alpha_1 v + 2u\alpha_2 - 2u\alpha_2 v - u - \alpha_1 v - \alpha_2 + \alpha_2 v)^m}$$

$(\alpha_2 v - \alpha_1 v - \alpha_2)^{m-i}(\alpha_1 v + \alpha_2 - \alpha_2 v - 1)^i$ can be expressed as $\sum\limits_{l_1=0}^{m} c_{l_1,i} B_{l_1}^m(v)$ by a linear matrix solving. Thus we have

$$B_i^m(s) = \frac{\sum\limits_{l_1=0}^{m} c_{l_1,i} B_{l_1}^m(v) B_i^m(u)}{(2u\alpha_1 v + 2u\alpha_2 - 2u\alpha_2 v - u - \alpha_1 v - \alpha_2 + \alpha_2 v)^m} \tag{9}$$

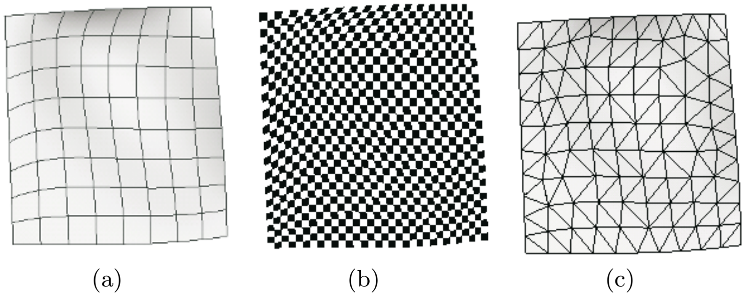


Fig. 4. Texture mapping and tessellating results of a Bézier surface: (a) Bézier surface and its optimal parameterization; (b) texture mapping result of reparameterized surface; (c) tessellating result by mapping Figure 1(d) to the reparameterized surface

Similarly, we have

$$B_j^n(t) = \frac{\sum_{l_2=0}^{n} a_{l_2,j} B_{l_2}^n(u) B_j^n(v)}{(2u\beta_1 v + 2v\beta_2 - 2v\alpha_2 u - v - \alpha_1 u - \alpha_2 + \alpha_2 u)^n} \tag{10}$$

From Equations (9) and (10), we obtain

$$B_i^m(s)B_j^n(t) = \frac{\sum_{l_1=0}^{m}\sum_{l_2=0}^{n} c_{l_1,i} a_{l_2,j} \frac{\binom{m}{l_1}\binom{n}{j}\binom{n}{l_2}\binom{m}{i}}{\binom{m+n}{l_1+j}\binom{m+n}{l_2+i}} B_{l_1+j}^{m+n}(v) B_{l_2+i}^{m+n}(u)}{D_1 D_2}$$

where $D_1 = (2u\alpha_1 v + 2u\alpha_2 - 2u\alpha_2 v - u - \alpha_1 v - \alpha_2 + \alpha_2 v)^m$ and $D_2 = (2u\beta_1 v + 2v\beta_2 - 2v\alpha_2 u - v - \alpha_1 u - \alpha_2 + \alpha_2 u)^n$. Thus the (k_1, k_2) control point of the reparameterized surface is

$$\frac{\sum_{i=\max(k_1-n,0)}^{k_1} \sum_{j=\max(k_2-m,0)}^{k_2} c_{k_2-j,i} a_{k_1-i,j} \frac{\binom{m}{k_2-j}\binom{n}{j}\binom{n}{k_1-i}\binom{m}{i}}{\binom{m+n}{k_2}\binom{m+n}{k_1}} \mathbf{P}_{i,j}}{\sum_{i=\max(k_1-n,0)}^{k_1} \sum_{j=\max(k_2-m,0)}^{k_2} c_{k_2-j,i} a_{k_1-i,j} \frac{\binom{m}{k_2-j}\binom{n}{j}\binom{n}{k_1-i}\binom{m}{i}}{\binom{m+n}{k_2}\binom{m+n}{k_1}}},$$

the weight of the (k_1, k_2) control point is

$$\sum_{i=\max(k_1-n,0)}^{k_1} \sum_{j=\max(k_2-m,0)}^{k_2} c_{k_2-j,i} a_{k_1-i,j} \frac{\binom{m}{k_2-j}\binom{n}{j}\binom{n}{k_1-i}\binom{m}{i}}{\binom{m+n}{k_2}\binom{m+n}{k_1}}.$$

Thus we get the reparameterized surface with degree $(m + n) \times (m + n)$.

3.2 Determining the Coefficients

The four coefficients α_1, α_2, β_1 and β_2 influence the parameterization of the resultant surface greatly. α_1 and α_2 affect the shape and distribution of v curves while the β_1 and β_2 affect the shape and distribution of u curves. To obtain a satisfying representation (more uniform parameter lines), we compute the α_i for some u curves $X(u, v_i)$ firstly. A least square technique is then applied to interpolate the α_i to get α_1 and α_2. β_1 and β_2 can be computed similarly. For most surface cases, the quadratic reparameterizations can produce more uniform parameter lines. For the surface shown in Figure 1(a), the rendering and tessellating results of the reparameterized surface using quadratic reparameterization are shown in Figure 4.

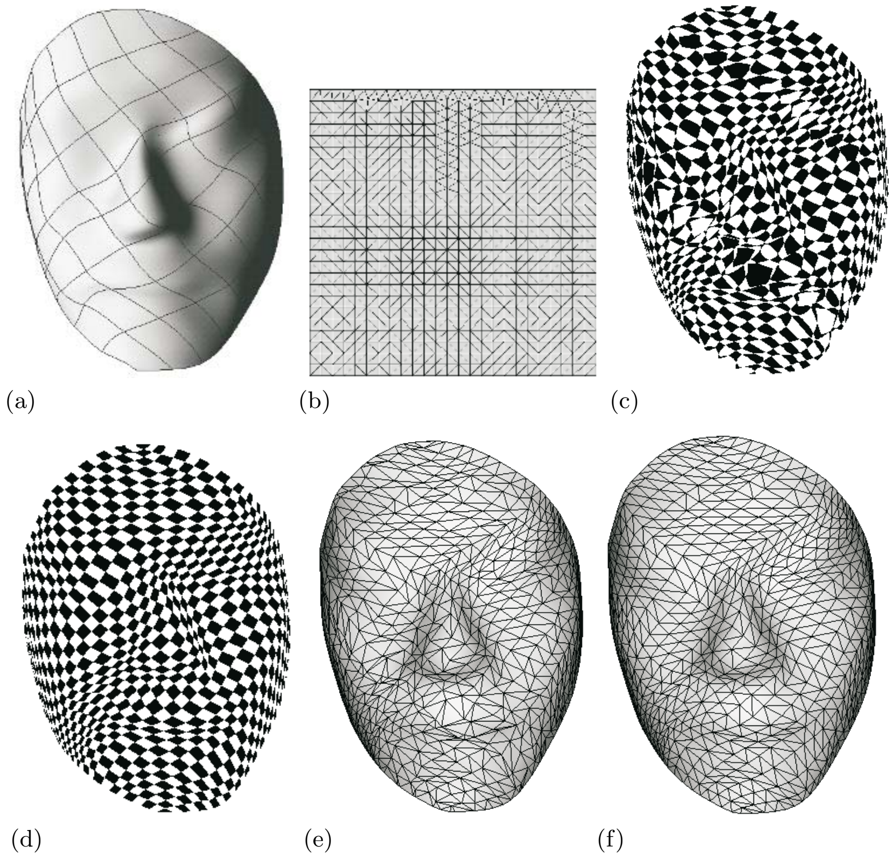


Fig. 5. Texture mapping and tessellating results of Bézier patches: (a) an arbitrary network of bicubic Bézier patches; (b) triangular mesh of the parameter domain; (c) texture mapping result of surfaces (a); (d) texture mapping result of the reparameterized patches; (e) tessellating result of (a) by mapping (b) to the patches; (f) tessellating result of (a) by mapping (b) to the reparameterized patches

From Figure 4, we can see that the quadratic reparameterization can change the shape of the parameter lines as well as the distribution of the parameter lines. The example shown in Figure 5 illustrates the application of quadratic reparameterization in animation industries. In Figure 5, we use a bicubic B-spline surface to approximate the original face model with 1024 triangles. Then the B-spline surface is split into 196 bicubic Bézier patches (see Figure 5(a)). After the quadratic reparameterization, the curves shared by two Bézier patches should have the same control points and the same weights. To achieve this, the coefficients of the quadratic reparameterization are determined by interpolating the reparameterization coefficients of the boundary curves directly. No interior u(or v) curves are involved in the computation. For the rendering and tessellating applications, the quadratic reparameterization produces more satisfying results with a cost that the degree of the surface patches is raised to $(m+n)\times(m+n)$.

4 Conclusions

We have shown that linear Möbius transformations can not change the shape of the parameter lines. What changes is the distribution of the parameter lines. In order to obtain more uniform parameter lines, a quadratic reparameterization algorithm is presented to reparameterize the Bézier surfaces. The examples indicate that, in practice, the algorithm produces significantly more uniform parameter lines across the Bézier surfaces.

Acknowledgements

The authors appreciate the comments and suggestions of the anonymous reviewers. The research was supported by Chinese 973 Program(2004CB719400), and the National Science Foundation of China (60403047). The second author was supported by the project sponsored by a Foundation for the Author of National Excellent Doctoral Dissertation of PR China (200342), and a Program for New Century Excellent Talents in University(NCET-04-0088).

References

1. Farouki RT. Optimal parameterizations. Computer Aided Geometric Design 1997,14(2):153-168.
2. Jüttler B. A vegetarian approach to optimal parameterizations. Computer Aided Geometric Design, 1997,14(9):887-890.
3. Costantini P, Farouki RT, Manni C and Sestini A. Computation of optimal composite re-parameterizations. Computer Aided Geometric Design 2001,18(9):875-897.
4. Yeh S-S, Hsu P-L. The speed-controlled interpolator for machining parametric curves. Computer-Aided Design 1999,31(5):349-357.
5. Yang DCH, Wang FC. A quintic spline interpolator for motion command generation of computer-controlled machines. ASME J. Mech. Design 1994,116:226-231.

6. Yang DCH, Kong T. Parametric interpolator versus linear interpolator for precision CNC machining. Computer-Aided Design 1994,26(3),225-234.
7. Wever U. Optimal parameterization for cubic spline. Computer-Aided Design 1991,23(9):641-644.
8. Wang FC, Yang DCH. Nearly arc-length parameterized quintic spline interpolation for precision machining. Computer-Aided Design 1993,25(5):281-288.
9. Wang FC, Wright PK, Barsky BA, Yang DCH. Approximately arc-length parameterized C^3 quintic interpolatory splines. ASME J. Mech. Design 1999,121:430-439.
10. Shpitalni M, Koren Y, Lo CC. Realtime curve interpolators. Computer-Aided Design 1994,26(11):832-838.
11. Ong BH. An extraction of almost arc-length parameterization from parametric curves. Ann. Numer. Math. 1996,3:305-316.
12. Piegl LA. and Richard AM. Tessellating trimmed NURBS surfaces. Computer-Aided Design 1995,27(1):15-26.
13. Ng WMM and Tan ST. Incremental tessellation of trimmed parametric surfaces. Computer-Aided Design 2000,32(4):279-294.
14. Hamann B, Tsai PY. A tessellation algorithm for the representation of trimmed nurbs surfaces with arbitrary trimming curves. Computer-Aided Design 1996,28(6/7):461-472.

Constrained Delaunay Triangulation Using Delaunay Visibility

Yi-Jun Yang[1,2], Hui Zhang[1], Jun-Hai Yong[1], Wei Zeng[3], Jean-Claude Paul[1], and Jiaguang Sun[1,2]

[1] School of Software, Tsinghua University, Beijing, China
[2] Department of Computer Science and Tech., Tsinghua University, Beijing, China
[3] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China

Abstract. An algorithm for constructing constrained Delaunay triangulation (CDT) of a planar straight-line graph (PSLG) is presented. Although the uniform grid method can reduce the time cost of visibility determinations, the time needed to construct the CDT is still long. The algorithm proposed in this paper decreases the number of edges involved in the computation of visibility by replacing traditional visibility with Delaunay visibility. With Delaunay visibility introduced, all strongly Delaunay edges are excluded from the computation of visibility. Furthermore, a sufficient condition for DT (CDT whose triangles are all Delaunay) existence is presented to decrease the times of visibility determinations. The mesh generator is robust and exhibits a linear time complexity for randomly generated PSLGs.

1 Introduction

Triangulation of a planar straight-line graph (PSLG) [1] plays an important role in surface visualization[2], stereolithography[3], garment design [4], pattern recognition [5], surface reconstruction [6,7,8] and finite element analysis [9,10,11]. Quality of the mesh strongly influences the accuracy and efficiency of design and analysis. Delaunay triangles have desirable properties such as maximum minimum angle and suitability for Delaunay refinement algorithms [12]. Delaunay triangulations[13](Fig. 1(b)) are convex. However, PSLGs usually are not. PSLGs have edge constraints that must be respected by the mesh (Fig. 1(a)). To address this problem, two methods have been established: conforming Delaunay triangulation and constrained Delaunay triangulation (CDT).

Conforming Delaunay approaches [14,15,16] repeatedly insert Steiner points into the mesh, while the mesh is Delaunay refined accordingly, until constraints are respected by the mesh: each edge is represented by a union of contiguous sequence of Delaunay triangulation edges. All triangles in the resulting mesh are Delaunay. The problem of where to insert points to obtain constraint conformity is difficult to solve. Edelsbrunner [14] shows that for any PSLG, there exists a Delaunay triangulation of O(m^2n) points that conforms to the PSLG where m, n are the number of constrained edges and the number of points in the PSLG respectively. Whether there exists a PSLG that really needs so many Steiner

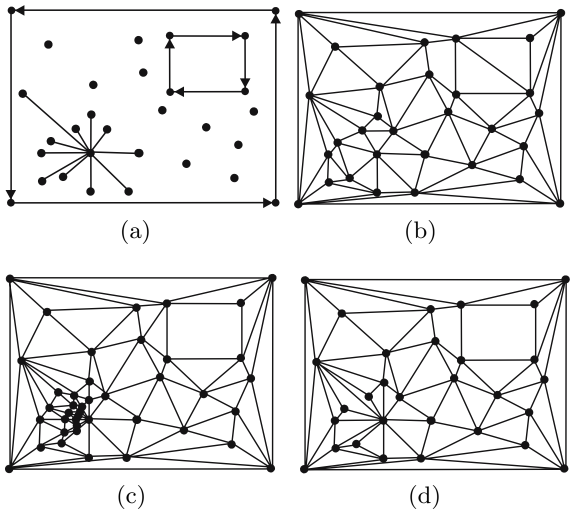


Fig. 1. A PSLG and its triangulations. (a) A PSLG (the interior square is a hole). (b) Delaunay triangulation. (c) Conforming Delaunay triangulation. (d) Constrained Delaunay triangulation.

points is not confirmed. But PSLGs for which O(mn) augmenting points are needed are known. It is an open problem to close the gap between O(m^2n) and O(mn) bound.

CDT [9,10,17,18,19,20]involves a set of edges and points while maintaining most of the favorable properties of Delaunay triangulation (such as maximum minimum angle). Compared with conforming Delaunay triangulation, Steiner points and unnecessary short edges are not involved in CDT. An example is given in Fig. 1 to illustrate the differences between conforming Delaunay triangulation and CDT. For the PSLG shown in Fig. 1(a), CDT (Fig. 1(d)) has no Steinter points involved compared with conforming Delaunay triangulation (Fig. 1(c)). Delaunay triangulation of the PSLG is also given (Fig. 1(b)) to illustrate the differences between Delaunay triangulation and CDT. Lee and Lin [10] present a divide and conquer algorithm with O($n$lg$n$) time complexity which has touched the lower bound of CDT. Bentley [21] uses a uniform grid to solve the nearest neighbor searching problem and proves that, under the assumption that points are chosen independently from a uniform distribution on the unit square in 2D, the nearest neighbor of a query point can be found by spiral search in constant expected time. Piegl[20] uses a uniform grid to accelerate two-dimensional constrained Delaunay triangulation for multiply connected polygonal domains. Exterior triangles generated during the triangulation process are removed from the output by a labeling procedure. Based on the uniform grid method, Klein[19] uses a divide and conquer paradigma to compute the constrained Delaunay triangulation of polygonal domains. No excess triangles are generated and the algorithm exhibits a linear time complexity for randomly generated polygons. However, extra operations are needed for multiply connected polygons.

Based on the uniform grid method, an advancing front algorithm to construct the constrained Delaunay triangulation of a PSLG is presented in this paper. In order to decrease the number of edges involved in the computation of visibility, Delaunay visibility is introduced. Two points are Delaunay visible from each other if the segment between the two points intersects no edges of the PSLG except strongly Delaunay edges, which are transparent for Delaunay visibility determinations. Furthermore, to decrease the times of visibility determinations, a sufficient condition for DT (CDT whose triangles are all Delaunay) existence is presented. Once the sufficient condition is satisfied during the triangulation process, no more visibility determinations are performed. The algorithm is fast and easy to implement.

This paper is organized as follows. In Section 2, some fundamental definitions used in this paper are presented. Section 3 describes how to construct constrained Delaunay triangles. Delaunay visibility as well as edge protection is presented in Sections 4 and 5 respectively. Section 6 gives some examples and analyzes the effects of Delaunay visibility and edge protection, followed by Section 7, which concludes the paper.

2 Fundamental Definitions

Definition 1 (Delaunay visibility)**.** Given a PSLG X, two points $p_1, p_2 \in P$ are Delaunay visible from each other if the segment $\widetilde{e}(p_1, p_2)$ intersects no edges $e \in E$ except strongly Delaunay edges. Two points $p_1, p_2 \in P$ that are Delaunay visible from each other are not always visible from each other as the segment $\widetilde{e}(p_1, p_2)$ may intersect some strongly Delaunay edges. An example is given in Fig. 2. There exists a circumcircle C of edge $e(p_3, p_4)$ which doesn't have any other points except points p_3 and p_4 in its interior or on its boundary. Then $e(p_3, p_4)$ is strongly Delaunay. Thus p_1 and p_2 are Delaunay visible from each other. However p_1 and p_2 are not visible from each other.

Definition 2 (Triangulation of a planar domain)**.** If the PSLG X is a planar domain, the triangulation of the planar domain is composed of triangles in T(X) that are inside the polygonal domain.

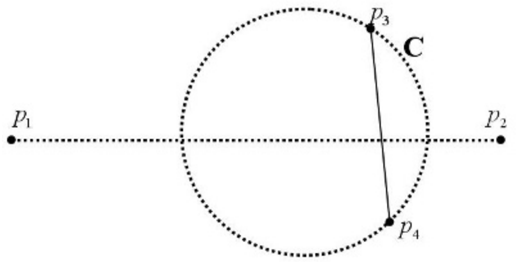


Fig. 2. p_1 and p_2 are Delaunay visible from each other, however not visible from each other

Definition 3 (Constrained Delaunay triangulation)**.** For any PSLG $X = (P, E)$, the CDT of X, denoted by CDT(X), is a triangulation T(X) in which each triangle is constrained Delaunay.

Definition 4 (Edge protection)**.** Given a PSLG $X = (P, E)$, X is edge protected if each edge $e \in E$ is Delaunay.

3 Constructing Constrained Delaunay Triangles

Before describing the details of constructing constrained Delaunay triangles, we define the orientation of triangles and edges. For a triangle denoted by $\triangle(p_1, p_2, p_3)$, its vertices p_1, p_2 and p_3 are listed in counterclockwise order. For a directed edge denoted by $e(p_1, p_2)$, the edge is oriented $p_1 \rightarrow p_2$ and the third point is searched for in the half plane to its left. If the PSLG is a polygonal domain, the boundary edges have a single orientation (thus, a triangle will be constructed only on their "inward" side) and internal edges have a double orientation (thus, triangles will be constructed on both sides). With the above orientation definitions, the constrained Delaunay triangles are constructed using the advancing front method as follows. First, all edges in E are designated as the front list. If the front list is not empty, the first directed edge in the current front list is selected as the active edge and a constrained Delaunay triangle is formed as follows. Let $\widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3})$ denote the region strictly inside the circumcircle $C(p_{i_1}, p_{i_2}, p_{i_3})$ of triangle $\triangle(p_{i_1}, p_{i_2}, p_{i_3})$. Then we have:

Lemma 1 [19]**.** For a constrained Delaunay edge $e(p_{i_1}, p_{i_2})$, triangle $\triangle(p_{i_1}, p_{i_2}, p_{i_3})$ is constrained Delaunay if and only if

(1) $p_{i_3} \in V_{i_1 i_2}$ where $V_{i_1 i_2} = \{p \in P | (\widetilde{e}(p_{i_1}, p) \cup \widetilde{e}(p_{i_2}, p)) \cap e_j = \varnothing, \forall e_j \in E - \{e(p_{i_1}, p), e(p_{i_2}, p)\}\}$. That is, if segments $\widetilde{e}(p_{i_1}, p_{i_3})$ and $\widetilde{e}(p_{i_2}, p_{i_3})$ are drawn, they have no intersections with constrained edges excluding the segments themselves.

(2) $\widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3}) \cap V_{i_1 i_2} = \varnothing$. That is, the circumcircle of $\triangle(p_{i_1}, p_{i_2}, p_{i_3})$ does not contain any point that is visible from p_{i_1} and p_{i_2}.

If a point satisfying Condition (1) is found, the point satisfying Conditions (1) and (2) of Lemma 1 can be obtained by repeatedly replacing p_{i_3} with an arbitrary point $p \in \widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3}) \cap V_{i_1 i_2}$. Given a PSLG $X = (P, E)$, all points and edges are put into a uniform grid to accelerate the Delaunay point search [19].

4 Delaunay Visibility

In Section 3, all edges $e \in E$ are involved in the visibility determinations between points $p \in P$. Though the grid structure method can constrain the intersection judgments in a local area around the current active edge, visibility determination is still time consuming. In order to decrease the number of edges involved in visibility determinations, Delaunay visibility is put forward.

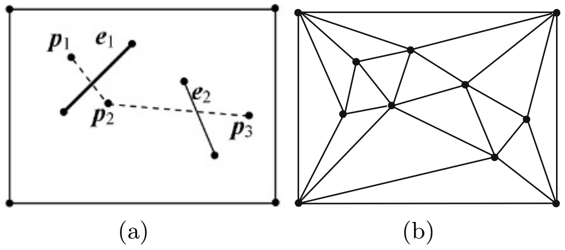


Fig. 3. Delaunay visibility (the bold line denotes an edge that is not strongly Delaunay). (a) A PSLG. (b) Delaunay triangulation.

In order to use the Delaunay visibility, we should identify the strongly Delaunay edges. Whether an edge is strongly Delaunay can be judged as follows. If an edge is involved in Delaunay triangulation of the PSLG, it is Delaunay, but not always strongly Delaunay. Thus symbolic perturbation [24] is introduced to simulate the circumstance where no four points are on a common circle. With perturbation involved, an edge is also strongly Delaunay if it is Delaunay. In Fig. 3(a), e_1 is not involved in the Delaunay triangulation while e_2 is involved in the Delaunay triangulation (as Fig. 3(b) shows). So e_1 is not strongly Delaunay and e_2 is strongly Delaunay. Since strongly Delaunay edges are apparent for Delaunay visibility determinations, p_2 and p_3 are Delaunay visible from each other while p_1 and p_2 are not Delaunay visible from each other. p_1 and p_2 as well as p_2 and p_3 are not visible from each other. In Section 3, all constrained edges are involved in the visibility determinations. If visibility is replaced with Delaunay visibility, all strongly Delaunay edges are excluded from Delaunay visibility determinations. The crucial problem brought is whether the CDT criteria is satisfied when visibility is replaced with Delaunay visibility. The following theorem gives a positive answer to this question.

Theorem 1. Let E_1 denote the strongly Delaunay edge set of the PSLG. For a constrained Delaunay edge $e(p_{i_1}, p_{i_2})$, triangle $\triangle(p_{i_1}, p_{i_2}, p_{i_3})$ is constrained Delaunay if and only if

(1) $p_{i_3} \in DV_{i_1 i_2}$, where $DV_{i_1 i_2} = \{p \in P | (\widetilde{e}(p_{i_1}, p) \cup \widetilde{e}(p_{i_2}, p)) \cap e_j = \varnothing, \forall e_j \in E - \{e(p_{i_1}, p), e(p_{i_2}, p)\} - E_1\}$. That is, the point P_{i_3} can be joined to p_{i_1} and p_{i_2} without intersecting any non-strongly Delaunay edges (excluding the segments themselves).

(2) $\widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3}) \cap DV_{i_1 i_2} = \varnothing$. That is, the circumcircle of $\triangle(p_{i_1}, p_{i_2}, p_{i_3})$ contains no points that are Delaunay visible from points p_{i_1} and p_{i_2}.

Proof. Theorem 1 can be understood as: if we delete strongly Delaunay edges from the constrained edge set of the PSLG X, the CDT is unchanged. This assertion can be demonstrated by the following statements. Let E_1 be the strongly Delaunay edge set of the PSLG X. Then we have $E_1 \subseteq E$. Also let $\widetilde{P}$ and $\widetilde{E}$ denote the point and edge set of the PSLG $\widetilde{X}$ derived from PSLG X by subtracting E_1 from the edge set. Thus we have $\widetilde{P} = P$ and $\widetilde{E} = E - E_1$. Given the CDT of

X denoted by $T(X)$, we will show that $T(X)$ is also a CDT for $\widetilde{X}$. Because $T(X)$ is a triangulation of the PSLG X, from Definition 7, we have $E_1 \subseteq E \subseteq ET(X)$ where $ET(X)$ denotes the edge set of $T(X)$. From Remark 14, we know that all edges of $ET(X)$ are either in E or locally Delaunay. Because $E_1 \subseteq E$, thus each edge of $ET(X) - E_1$ is either in $E - E_1$ or locally Delaunay. Because each edge of E_1 is strongly Delaunay, from Remark 9, every edge is locally Delaunay. Thus we have that each edge of $ET(X)$ is either in $E - E_1 = \widetilde{E}$ or locally Delaunay. From Remark 14, $T(X)$ is the CDT of PSLG $\widetilde{X}$. $\square$

From Theorem 1, strongly Delaunay edges are excluded from visibility determinations. If E_1 in Theorem 1 is replaced with set $E_2 \subset E_1$, Theorem 1 still holds. With Delaunay visibility introduced, another problem that must be handled is how to judge whether an edge is Delaunay. If the judgments are time consuming compared with the merit introduced by Delaunay visibility, Delaunay visibility is meaningless. Luckily, we mainly use the information brought by the construction of constrained Delaunay triangles. The details are shown as follows. First, the initial front list is just composed of directed edges of E, which is administrated by a linked list. New generated edges are appended to the list. Each time the first undelaunay edge in the list is selected as the active edge. Thus edges of E are processed first. Whether an edge of E is Delaunay can be judged after the derived triangle is formed. During the construction of constrained Delaunay triangles, we should judge whether $\widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3}) \cap DV_{i_1 i_2} = \varnothing$ holds. Any point $p \in \widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3}) \cap P$ is tested whether it is Delaunay visible from the active edge. Thus whether $\widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3}) \cap P = \varnothing$ holds is obtained by the way. If $\widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3}) \cap P = \varnothing$ holds for the active edge $e(p_{i_1}, p_{i_2})$, $e(p_{i_1}, p_{i_2})$ is a Delaunay edge. Then eliminate $e(p_{i_1}, p_{i_2})$ from edge lists of the uniform grid cells. If there exists some empty circumcircle (not necessarily $\widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3})$), edge $e(p_{i_1}, p_{i_2})$ is Delaunay. So $\widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3}) \cap P = \varnothing$ is only a sufficient but not necessary condition for edge $e(p_{i_1}, p_{i_2})$ to be Delaunay.

5 Edge Protection

Shewchuk [12] presents edge protection to guarantee the existence of CDT in 3D. If edge protection is introduced to CDT in 2D, we will get a sufficient and necessary condition for DT (CDT whose simplexes are all Delaunay) existence. Given a PSLG, we have:

Theorem 2. A PSLG X has a DT if and only if X is edge protected.

Proof of theorem 2 follows directly from the definition of edge protection (Definition 4). If all edges in the current front list are Delaunay, the front (a domain bounded by all current front edges) is edge protected. Then from Theorem 2, the front has a DT. No visibility determinations are needed to triangulate the interior of the front. Now the test of edge protection is the most crucial problem for the application of Theorem 2. It is the same as for the application of

Theorem 1. We use the information brought by the construction of constrained Delaunay triangles. Few extra operations are introduced in this approach. To use this approach, the front advances further than necessary . Now we give the main flow of our algorithm. There are two main loops in the algorithm. The first loop triangulates all edges labeled undelaunay in the front. After the first loop, all edges in the current front list are Delaunay. Thus no visibility determinations are needed during the second loop processing. Given a PSLG, the following steps are performed.

1. Add all directed edges of E to the initial front list and label all edges as undelaunay. Calculate the size of E and assign it to the variable *number*. Assign $\varnothing$ to E_1. Select the first edge of the initial front list as active edge.
2. For the active edge $e(p_{i_1}, p_{i_2})$, find the Delaunay point p_{i_3} with the following properties:
   (1) $p_{i_3} \in DV_{i_1 i_2}$,where $DV_{i_1 i_2} = \{p \in P | (\widetilde{e}(p_{i_1}, p) \cup \widetilde{e}(p_{i_2}, p)) \cap e_j = \varnothing, \forall e_j \in E - \{e(p_{i_1}, p), e(p_{i_2}, p)\} - E_1\}$;
   (2) $\widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3}) \cap DV_{i_1 i_2} = \varnothing$.
   That is, Step 2 finds a point p_{i_3} for the active edge such that p_{i_3} can be joined to the two endpoints p_{i_1}, p_{i_2} of the active edge without intersecting any of the constrained edges excluding identified Delaunay edges, and forms with them a triangle, whose circumcircle contains no points that are Delaunay visible from p_{i_1} and p_{i_2} .
3. If $e(p_{i_1}, p_{i_2}) \in E$ and $\widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3}) \cap P = \varnothing$, label $e(p_{i_1}, p_{i_2})$, $e(p_{i_1}, p_{i_3})$ and $e(p_{i_2}, p_{i_3})$ as Delaunay and perform $E_1 = E_1 \cup (\{e(p_{i_1}, p_{i_2}), e(p_{i_1}, p_{i_3}), e(p_{i_2}, p_{i_3})\} \cap E)$. That is, if the circumcircle of formed triangle in Step 2 has no points inside, each edge of the triangles is labeled as Delaunay, and added to Delaunay edge set E_1 if it is a constrained edge. Update front and the variable *number* according to two generated edges.
4. Choose the first edge labeled undelaunay from the current front as active edge and repeat Steps 2-4 until the variable *number* becomes zero.
5. Choose the first edge from the current front as active edge.
6. For the active edge $e(p_{i_1}, p_{i_2})$, find the third point p_{i_3} with the following properties:
   (1) p_{i_3} lies in the left half plane to $e(p_{i_1}, p_{i_2})$;
   (2) $\widetilde{C}(p_{i_1}, p_{i_2}, p_{i_3}) \cap P = \varnothing$.
   That is, Step 6 finds a point p_{i_3} for the active edge such that circumcircle of the formed triangle $\triangle(p_{i_1}, p_{i_2}, p_{i_3})$ contains no points.
7. Update the front according to two generated edges.
8. Choose the first edge from the current front as active edge and repeat Steps 6-8 until the front list is empty.

Determinations of Delaunay visibility are needed only when edges labeled undelaunay are processed. Each time an edge labeled undelaunay is processed, a new triangle is generated. Front and the variable *number* are updated accordingly. Whenever *number* becomes zero, the current front can be triangulated without determinations of Delaunay visibility (as Step 6 shows).

6 Results

Three examples of CDT are given in Fig. 4. Time comparison between proposed algorithm without symbolic perturbation with Klein's algorithm [19] and Piegl's algorithm [20] is listed in Table 1.

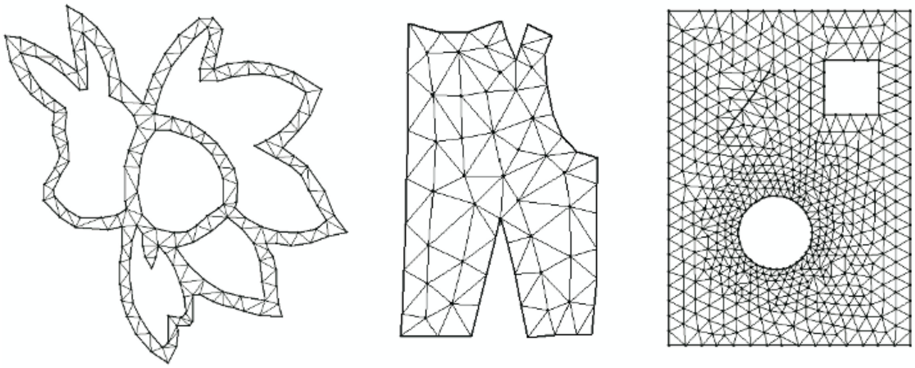


Fig. 4. Three examples of CDT

Table 1. Time (ms) comparison between proposed algorithm without symbolic perturbation with Klein's algorithm and Piegl's algorithm for randomly generated polygons of different size

Size	200	400	600	800	1000	1200	1400	1600	1800	2000
Piegl's algorithm	24	62	143	179	228	272	304	375	418	592
Klein's algorithm	17	33	51	69	87	103	118	135	153	172
Our algorithm	5	10	16	22	26	29	35	38	44	53

It can be seen that the proposed algorithm, CDT utilizing DV (Delaunay visibility) and EP (edge protection), is faster than Klein's algorithm and Piegl's algorithm. Furthermore, the proposed algorithm exhibits a linear time complexity for randomly generated polygons (see Table 1). Delaunay visibility can be used extensively to accelerate two-dimensional CDT algorithms [5,10,19]. Edge protection can be applied for CDT algorithms utilizing advancing front or shelling method [5,11,20]. Introduction of symbolic perturbation [24] can handle degeneracies, thus simplify the programming and enhance robustness of our algorithm. So the time listed in the following tables is measured for presented algorithm with perturbation involved.

All above algorithms are run in the environment with Intel Pentium IV CPU 2.0GHz, 256Mb, Microsoft Windows 2000 and Microsoft Visual C++ 6.0.

7 Conclusions

A fast and easy to implement algorithm is presented for CDT construction of a PSLG. Visibility determinations between the points of a PSLG are time consuming. In order to decrease the number of edges involved in visibility determinations, Delaunay visibility is presented. Furthermore, in order to decrease the times of visibility determinations between the points of a PSLG, a sufficient condition for DT is presented and used combined with uniform grid and advancing front techniques in our algorithm.

Acknowledgements

The authors appreciate the comments and suggestions of the anonymous reviewers. The research was supported by Chinese 973 Program(2004CB719400), and the National Science Foundation of China (60533070). The third author was supported by the project sponsored by a Foundation for the Author of National Excellent Doctoral Dissertation of PR China (200342), and a Program for New Century Excellent Talents in University(NCET-04-0088).

References

1. Ronfard RP and Rossignac JR. Triangulating multiply-connected polygons: A simple, yet efficient algorithm. Computer Graphics Forum, 13(3):281-292, 1994.
2. Rockwood A, Heaton K and Davis T. Real-time rendering of trimmed surfaces. Computers & Graphics, 23(3): 107-116, 1989.
3. Sheng X and Hirsch BE. Triangulation of trimmed surfaces in parametric space. Computer-Aided Design, 24(8):437-444, 1992.
4. Obabe H, Imaoka H, Tomiha T and Niwaya H. Three dimensional apparel CAD system. Computer & Graphics, 26(2): 105-110, 1992.
5. Zeng W, Yang CL, Meng XX, Yang YJ and Yang XK. Fast algorithms of constrained Delaunay triangulation and skeletonization for band-images. In SPIE Defense and Security Symposium, 5403: 337-348, 2004.
6. Gopi M, Krishnan S and Silva CT. Surface reconstruction based on lower dimensional localized Delaunay triangulation. Computer Graphics Forum, 19(3):467-478, 2000.
7. Nonato LG, Minghim R, Oliveira MCF, Tavares G. A novel approach for Delaunay 3D reconstruction with a comparative analysis in the light of applications. Computer Graphics Forum, 20(2):161-174, 2001.
8. Marco A and Michela S. Automatic surface reconstruction from point sets in space. Computer Graphics Forum, 19(3):457-465, 2000.
9. Ho-Le K. Finite element mesh generation methods: a review and classification. Computer-Aided Design, 20(1):27-38, 1988.
10. Lee DT and Lin AK. Generalized Delaunay triangulations. Discrete & Computational Geometry, 1:201-217, 1986.
11. Yang YJ, Yong JH and Sun JG. An algorithm for tetrahedral mesh generation based on conforming constrained Delaunay tetrahedralization. Computers & Graphics, 29(4):606-615, 2005.

12. Shewchuk JR. Constrained Delaunay tetrahedralizations and provably good boundary recovery. In Eleventh International Meshing Roundtable, 193-204, 2002.
13. Cignoni P, Montani C, Perego R and Scopigno R. Parallel 3D Delaunay triangulation. Computer Graphics Forum, 12(3):129-142, 1993.
14. Edelsbrunner H and Tan TS. An Upper Bound for Conforming Delaunay Triangulations. Discrete & Computational Geometry, 10(2):197-213, 1993.
15. Nackman LR and Srinivasan V. Point placement for Delaunay triangulation of polygonal domains. In Proceeding of Third Canadian Conference Computational Geometry, 37-40, 1991.
16. Saalfeld A. Delaunay edge refinements. In Proceeding of Third Canadian Conference on Computational Geometry, 33-36, 1991.
17. Chew LP. Constrained Delaunay triangulations. Algorithmica, 4:97-108, 1989.
18. George PL and Borouchaki H. Delaunay Triangulation and Meshing: Application to Finite Elements. Paris: Hermes, 1998.
19. Klein R. Construction of the constrained Delaunay triangulation of a polygonal domain. In CAD Systems Development, 313-326, 1995.
20. Piegl LA and Richard AM. Algorithm and data structure for triangulating multiply connected polygonal domains. Computer & Graphics, 17(5): 563-574, 1993.
21. Bentley JL, Weide BW and Yao AC. Optimal expected-time algorithms for closest point problems. ACM Transactions on Mathematical Software, 6(4):563-580, 1980.
22. Devillers O, Estkowski R. Gandoin PM, Hurtado F, Ramos P and Sacristan V. Minimal Set of Constraints for 2D Constrained Delaunay Reconstruction. International Journal of Computational Geometry and Applications, 13(5):391-398, 2003.
23. Fang TP and Piegl LA. Delaunay triangulation using a uniform grid. IEEE Computer Graphics and Applications, 13(3):36-47, 1993.
24. Edelsbrunner H and Mcke EP. Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. ACM Transactions on Graphics, 9(1):66-104, 1990.

Immersing Tele-operators in Collaborative Augmented Reality

Jane Hwang[1], Namgyu Kim[1], and Gerard J. Kim[2,*]

[1] Department of Computer Science and Engineering
POSTECH, Pohang, Korea
[2] Department of Computer Science and Engineering
Korea University, Seoul, Korea
gjkim@korea.ac.kr

Abstract. In a collaborative system, the level of co-presence, the feeling of being with the remote participants in the same working environment, is very important for natural and efficient task performance. One way to achieve such co-presence is to recreate the participants as real as possible, for instance, with the 3D whole body representation. In this paper, we introduce a method to recreate and immerse tele-operators in a collaborative augmented reality (AR) environment. The method starts with capturing the 3D cloud points of the remote operators and reconstructs them in the shared environment in real time. In order to realize interaction among the participants, the operator's motion is tracked using a feature extraction and point matching (PM) algorithm. With the participant tracking, various types of 3D interaction become possible.

1 Introduction

Remote collaboration has been one of the key applications of shared virtual environments [1]. Mixed (or augmented) reality presents even more potential for remote collaboration, for instance, if it would be possible to recreate the remote operators in the real environment. In this situation, contrast to the virtual environment, both the operating environment and the participants are (or look) real, rather than graphical representations. Recreating and immersing the remote operators would undoubtedly improve the co-presence (the feeling of being with the remote participants in the same working environment [2]) among the collaborators and improve the efficiency of the group task performance.

In this paper, we introduce a method to recreate and immerse tele-operators in a collaborative augmented reality (AR) environment in a realistic manner, e.g. with 3D whole body representation. The two main issues in this research are (1) capturing and presenting a natural and photo-realistic whole body representation of the participants to the shared users, and (2) tracking participants motion (at least partially) for realizing 3D interaction among the users, all in real time. Our method starts with capturing

* Corresponding author.

the 3D cloud points of the remote operators using a 3D camera. The captured 3D clouds can be displayed in the shared augmented environment at a designated or marker tracked locations. Then, the operator's motion is partially tracked using a feature extraction and point matching (PM) algorithm. The feature extraction algorithm first finds important body features (or points) from the 3D camera and based on this information, the point matching algorithm is iteratively run to track them in real time. In order to start the point matching algorithm and converge to a good solution, the initial pose of the actor is estimated by inverse kinematics. Because a 3D camera is used for capturing the 3D point clouds, the users can have a limited range of views, rather than an image with a single point of view. In summary, the system provides a natural looking imagery (e.g. video captured whole body participants registered in a real environment) resembling a situation in which all the participants are *present* and *interact-able* within the same environment.

This paper is organized as follows. First, we review previous research related to this work in section 2. In section 3, we present each important step in capturing, displaying and reconstructing 3D information of the remote participants for the shared augmented environment. Finally, we conclude the paper with an executive summary and future directions.

2 Related Work

One of the typical remote collaborative systems is tele-conferencing. Tele-conferencing naturally focuses only on the exchange of words (voice) and 2D video rather than physical interaction [3]. In terms of providing the presence of the participants, conventional tele-conferencing systems only offer imagery of faces or upper body with a single view point. Several works have addressed this problem by employing eye or motion tracking and image warping to provide a more natural platform for conversation with mutual gaze [4, 5]. However, most teleconference systems are still 2D based. Hauber et al. have developed an AR based teleconference system in which 2D video objects were placed (or registered) in the real (3D) environment [6]. However, the 2D videos were taken from a single view camera showing only faces of the participants.

On the other hand, shared networked virtual reality systems have been a popular platform for remote collaborative systems [7, 8, 9, 10]. VR based remote collaboration frameworks have the strength in enabling 3D physical collaborative tasks among the shared users. However, it has limitations in providing photo-realism of the participants and the working environment, and natural avatar control. Many researches have pointed to the importance of co-presence as an important factor in improving collaborative task performance [11], and identified, among others, the role of gaze, body gestures and whole body interaction as key elements [12].

Not too many attempts for remote collaboration have been made in the augmented reality area. Most AR based collaboration systems have been on-site (i.e. participants are all locally present) and focused on augmenting the real world to assist the collaboration [13]. Even though the participants are physically present, the imagery is still not natural, because the collaborators are seen wearing sensors and head mounted displays. Prince et al. has developed a system called the "3D-Live" [14]. In this work, a participant was captured by a multi-camera system and can be placed on a hand-held marker to be manipulated. While it has been reported such a style of teleconferencing has improved the social presence compared to the traditional audio and video teleconferencing interfaces [15], natural physical 3D interaction is still not possible and the participants are not life-sized as in the real world.

3 Immersing the Tele-operator

Fig. 1 shows the overall framework for immersing tele-operators (the client in the left part of the figure) in the shared augmented workspace (right part of the figure). There are three main steps to register the client co-worker into the shared environment, namely, actor extraction, actor tracking, and reconstruction. In the actor extraction stage, the client's 3D cloud data is obtained using a 3D camera and important feature points are extracted and analyzed. The analyzed data is used as a basis for feature point tracking in the second stage. The 3D cloud and motion data of the client are compressed and transmitted to the local workspace where the collaboration is needed. At the local workspace, the 3D cloud and motion data are used to display and reconstruct the client and allow 3D interaction. The procedure of extracting actor was similar to that of our previous research [16].

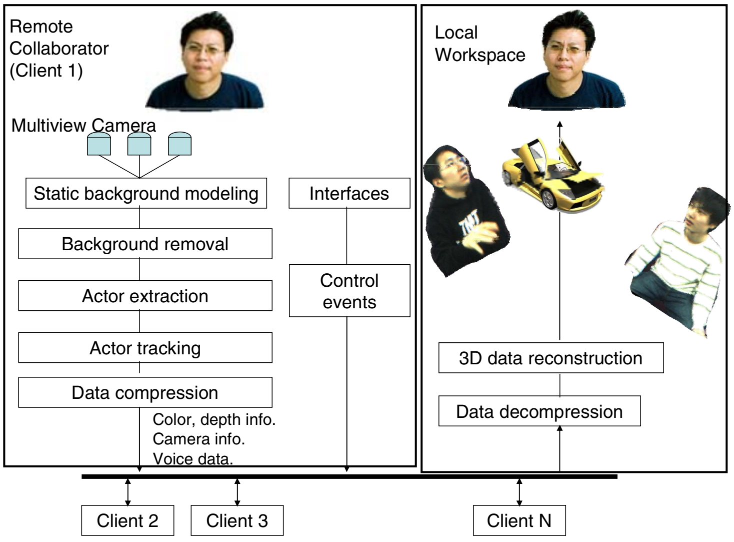


Fig. 1. The overall framework for immersing tele-operator in the shared AR workspace

3.1 Actor Extraction

We use a multi-view camera to capture color and 3D depth information simultaneously in real-time. The multi-view camera that we use is called the Digiclops™ and provides color and depth information in real time (at about 19 fps in a resolution of

320 X 240). The disparity estimation of the multi-view camera is based on a simple intensity-based pixel matching algorithm.

We have implemented a color-based object segmentation algorithm for extracting the actor (client) from the video image. We separate the actor from the static background by subtracting the current image from a reference image (i.e. the static background). The subtraction leaves only the non-stationary or new objects. The proposed segmentation algorithm consists of the following three steps: (1) static background modeling, (2) moving object segmentation, and (3) shadow removal. After segmentation, we correct the final image by, for instance, filling in holes.

3.1.1 Static Background Modeling

To separate moving objects (e.g. actor/client) from the natural background, we exploit the color properties and statistics of the background image. We first set up the multi-view camera with known extrinsic (estimated with initially measured values) and internal (estimated with values given by the camera specification) camera parameters. Then, the color statistics over a number of static background frames, *N(m, σ)* where *m* and *σ* denote the pixel-wise mean and standard deviation of the image, are collected. Let an arbitrary color image be denoted as *I(R,G,B)*. The mean image, $I_m(R,G,B)$, is used as the reference background image and the standard deviation, $I_\sigma(R,G,B)$, is used for a threshold based extraction of the actor (moving object). The mean image and standard deviation are calculated as follows:

$$I_m(R,G,B) = \frac{1}{L} \cdot \sum_{t=0}^{L-1} I_t(R,G,B) \tag{1}$$

$$I_\sigma(R,G,B) = \sqrt{\frac{1}{L} \cdot \sum_{t=0}^{L-1} (I_t(R,G,B) - I_m(R,G,B))^2} \tag{2}$$

where, *L* denotes the total number of image frames used to estimate the statistics. We empirically choose *L* to be 30.

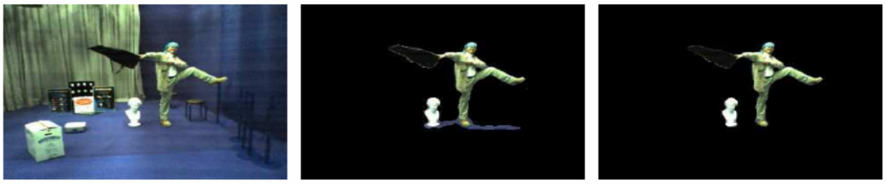

(a) Original Image (b) Object segmentation (c) Shadow Removal

Fig. 2. Moving object segmentation

3.1.2 Moving Object Segmentation

Each pixel can be classified into objects or background by evaluating the difference between the reference background image and the current image in the color space in terms of (*R,G,B*) (Fig. 2). To separate pixels of the moving objects from pixels of the background, we measure the Euclidian distance in the RGB color space, and then compare it with the threshold. The threshold is adjusted in order to obtain a desired detection rate in the subtraction operation.

3.1.3 Shadow Removal

Normally, the moving objects make shadows and shading effects according to (changing) lighting conditions. However, the RGB color space is not a proper space to deal with the (black and white) shadow and the shading effects. Instead, we use the normalized color space, which represents the luminance property better. In the normalized color space, we can separate the shadows from the background. The difference between the normalized color image $I_t(r,g,b)$ and the normalized mean of the reference image $I_m(r,g,b)$ is calculated. If the difference of the pixel is less than a threshold, the pixel is classified as part of the background. Note that we can speed up the process by only checking the difference in the normalized color space, if the pixel was classified as a part of the objects. If the pixel is classified as objects (in the normalized color space), then we further decompose the color difference into the brightness and chromaticity components. Then, based on the observation that shadow has similar chromaticity but slightly different (usually lower) brightness than that of the same pixel in the background image, we can label the pixels into shadows. We also exploit the depth information to reduce misclassification, while segmenting out the moving object. The underlying assumption is that the moving objects have a limited range of depth values (*i.e.* relatively small or thin). Using this assumption, we can remove spot noises in the segmented object. In addition, we apply two types of median filters (5 x 5 and 3 x 3) to fill the hole while maintaining sharp boundaries.

3.1.4 Depth Correction

Often the multi-view camera exhibits errors in the depth estimation. Therefore, we implemented a depth recovery algorithm similar to the method of hierarchical block matching. That is, for given a pixel, we search its neighborhood depth information on a 16 x 16 pixel block. The mean value of available neighborhood disparity values is set to disparity of the pixel.

3.1.5 Data Compression and Transmission

Even though the capture client 3D data is much less than what was originally captured by the 3D camera (whole scene), for faster network transmission and scalability, the client 3D data is compressed. The color image of the actor has a size of *(number of pixels considered as actor)* x *(rgb(3 bytes))* bytes and the size of the depth information is (*number of pixels considered as actor*) bytes (because depth can be expressed using gray level after 8-bit quantization). In total, each participant will result in approximately *(number of pixels considered as actor)* x *4* bytes. We used the JPEG compression method, and Table 1 shows the reduced amount data by the compression.

Table 1. Decreased data size after JPEG compression (KB/frame)

Raw Data / Actor (Worst Case)	100% JPEG	90% JPEG	80% JPEG
< 307 KB	< 87 KB	< 27 KB	< 16 KB

3.2 Actor Tracking

The heart of our actor tracking is in the use of the point matching (PM) algorithm as our system uses 3D point clouds as raw input. PM algorithm calculates body poses of a client by matching the 3D point clouds (as segmented out from the steps described in 3.2) and a pre-defined and appropriately scaled skeleton geometry (a 3D model made of cylinder, boxes and ellipsoids). While the pre-defined and scaled geometric model is not a true reflection of the actual client, such an approximation still proves effective in realizing 3D interaction without wired sensors. In order to apply the iterative PM algorithm efficiently, the initial pose of the client (or 3D model) is estimated by inverse kinematics specified by strategic body points identified by feature extraction.

3.2.1 Feature Finding

After background extraction, with the segmented client image, we can compute its contour on the mask image sequences[17]. By analyzing this contour image, we can infer lots of useful information such as the position of body center, head, hands and feet (Fig. 3).

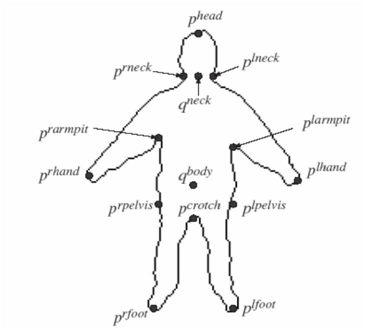


Fig. 3. Feature points on the contour

Fig. 3 shows all the feature points. Furthermore, we can extract regions of the body parts based on these feature points. For example, to find a left arm region, we trace the contour points from the left arm pit point to the left neck point. Then, we make a minimally enclosing rectangle for the traced contour points. Similarly, we define six body part regions for the left and right arms (trace arm pit to neck), left and right legs (from crotch to pelvis), chest (polygonal region bounded by next, arm pit and two closest points from center of mass of contour image) and head (from left neck to right neck) (See Fig. 4).

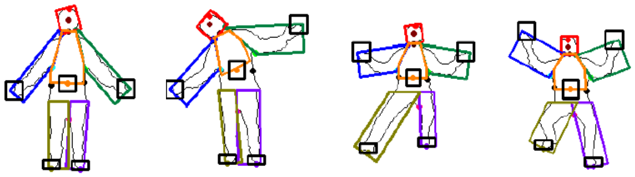


Fig. 4. Region of each body parts based on finding features

3.2.2 Base Human Modeling

To apply the PM algorithm and estimate the client's motion, there must be a 3D model in the first place and it must be scaled appropriately according to the client's physical dimensions. We developed an ellipsoid based human modeling tool for this purpose and scaled the 3D model according to the Korean Standard Body Size Index database (KSBSI) [18]. The age and height of the client is used as input and a appropriately sized 3D human model with a pre-defined skeleton structure is generated.

3.2.3 Point Matching

In our work, we applied a well known point matching algorithm called the Iterative Closet Point algorithm (ICP). ICP finds a transformation matrix that aligns two sets of 3D points [19] and involves the following procedures. Given correct correspondences, the ICP offers an algebraic solution (i.e. no iteration is needed). However, when the correspondence information is only approximated as in our case, ICP is executed iteratively. More specifically, ICP iteratively minimizes an error metric established between two sets of corresponding 3D point sets. Thus, it is vital to establish good correspondences between two point sets in the first place for fast convergence to a reasonable pose.

Given the extracted features and regions by the process described in 3.2.1, it is possible to approximate the 3D poses and positions of the coordinate systems of the skeleton (and thus the limbs too) using inverse kinematics (IK). The actual correspondence among two points (one from 3D cloud segmented as a certain part in the body and the other from the points that belong to the corresponding limb) is made based on the closest distance measure. For fast computation, a method called cyclic coordinate descent (CCD) method was used [20]. Observe that the inverse kinematics alone may generate unnatural poses that does not match that of the actual user as illustrated in Fig. 5 (left, lower legs). With the application of the ICP, a more correct body tracking is achieved (right, lower legs). The subsequent application of the ICP algorithm overcomes this problem by quickly registering the 3D point cloud to the predefined 3D human model.

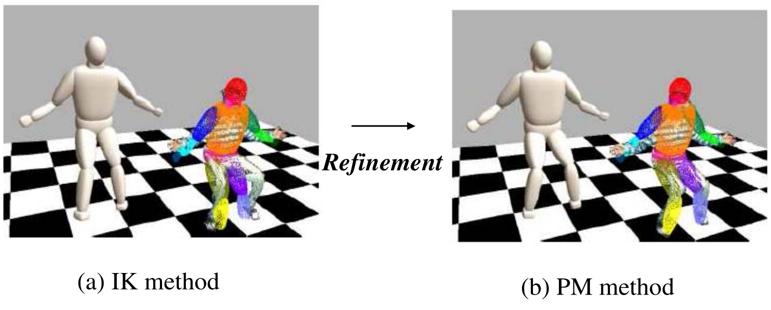


Fig. 5. Pose estimation based on IK and PM methods

3.3 Reconstruction: Display and Interaction in AR Environment

After receiving the client 3D model and motion, a client representation is reconstructed in the shared AR environment. The segmented image data of the client is decompressed and restored at a designated location within the AR environment. We use a marker to register the client into the AR environment. The marker tracking was realized by using the ARToolKit [21]. Fig. 6 shows the implementation result. Note that as in the 3D-Live system, the participant can be moved freely and offers limited view points.

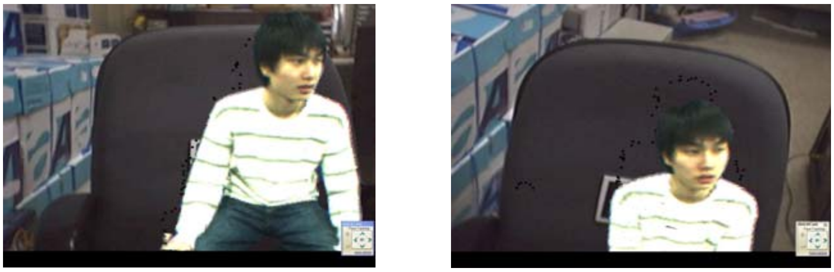


Fig. 6. A Reconstructed remote user in the local AR workspace: view from front (left) and view from top (right)

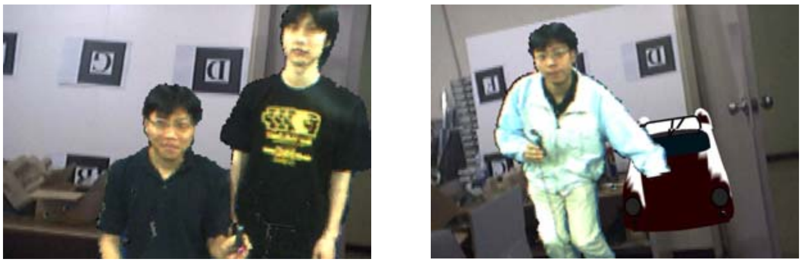


Fig. 7. Multiple participants in the shared AR environment (left). A remote user interacting with a virtual 3D car (right)

Fig 7 shows a case of multiple users and a virtual object all situated in the same workspace. With the 3D information of the clients, 3D interaction is possible as illustrated in the right part of Fig. 7 with the hand of the client inserted into the virtual car.

4 Conclusion

In this paper, we have presented a method for capturing and immersing a 3D tele-operator into a shared AR environment. The proposed system not only provides a life sized video based representation of the participants, but also their approximate 3D information for closer physical interaction. The system runs in real time on commodity computing and network hardware (although it requires a 3D camera) and is expected to be scalable to support more participants. We believe that the provision of life-sized and 3D representation of the participants is a key to making remote collaborative AR an effective platform for improved level of presence, and shared task performance. The proposed system can find many applications such as in medicine, training and entertainment. We are planning to further formally investigate its usability and scalability.

Acknowledgement

This work has been supported by the ITRC program of the Ministry of Information and Communication of Korea.

References

1. Lehner, V. D. and DeFanti, T. A.: Distributed Virtual Reality: Supporting Remote Collaboration in Vehicle Design, IEEE Computer Graphics and Applications, vol. 17, (1997) 13 - 17
2. Tromp, J., et al.: Small Group Behavior Experiments in the Coven Project, IEEE Computer Graphics and Applications, vol. 18, (1998) 53-63
3. Egido, C.: Video Conferencing as a Technology to Support Group Work: a Review of Its Failures. In Proceedings of the ACM Conference on Computer-supported Cooperative Work, Portland, Oregon, USA, (1988)
4. Buxton, W. A. S.: The Three Mirrors of Interaction: a Holistic Approach to User Interfaces. In Proceedings of the Friend21 '91 International Symposum on Next Generation Human Interface, Tokyo, Japan, (1991)
5. Yang, R. and Zhang, Z.: Eye Gaze Correction with Stereovision for Video-Teleconferencing, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, (2004) 956-960
6. Hauber, J., et al.: Tangible Teleconferencing. In Proceedings of the Sixth Asia Pacific Conference on Human Computer Interaction (APCHI 2004), Rotorua, New Zealand, (2004)
7. Kauff, P. and Schreer, O.: An Immersive 3D Video-conferencing System using Shared Virtual Team User Environments. In Proceedings of the International Conference on Collaborative Virtual Environments (CVE'02), (2002)
8. Mortensen, J., et al.: Collaboration in Tele-Immersive Environments. In Proceedings of the Eighth Eurographics Workshop on Virtual Environments, (2002)
9. Schreer, O. and Sheppard, P.: VIRTUE - The Step Towards Immersive Tele-Presence in Virtual Video Conference Systems. In Proceedings of the eWorks, Madrid, Spain, (2000)
10. Normand, V., et al.: The COVEN project: exploring applicative, technical and usage dimensions of collaborative virtual environments, Presence: Teleoperator and Virtual Environments, vol. 8, (1999) 218-236
11. Biocca, F., et al.: Toward a More Robust Theory and Measure of Social Presence: Review and Suggested Criteria, Presence: Teleoperators and Virtual Environments, vol. 12, (2003) 456 - 480
12. Garau, M., et al.: The Impact of Eye Gaze on Communication using Humanoid Avatars. In Proceedings of the SIGCHI conference on Human factors in computing systems, Seattle, Washington, USA, (2001)

13. Broll, W., et al.: The Virtual Round Table - a Collaborative Augmented Multi-User Environment. In Proceedings of the International Conference on Collaborative Virtual Environments (CVE'00), San Francisco, CA, USA, (2000)
14. Prince, S., et al.: 3D Live: Real Time Captured Content for Mixed Reality. In Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR'02), (2002)
15. Billinghurst, M., et al.: Real World Teleconferencing, IEEE Computer Graphics and Applications, vol. 22, (2002) 11-13
16. Kim, N., Woo, W., Kim, G. J., Park, C. M.: 3-D Virtual Studio for Natural Inter-"Acting". IEEE Trans. on Systems, Man, and Cybernetics, vol. 36, (2006) 758-773
17. Kass, M., et al.: Snakes: Active Contour Models. International Journal of Computer Vision, vol. 1, pp. 321-331, (1988)
18. Korean Body Size Index, Available at http://www.standard.go.kr.
19. Rusinkiewicz, S. and Levoy, M.: Efficient Variants of the ICP Algorithm. In Proceedings of the Third International Conference on 3D Digital Imaging and Modeling, (2001)
20. Wang, L. and Chen, C.: A combined optimization method for solving the inverse kinematics problem of mechanical manipulators, IEEE Trans. on Robotics and Applications, vol. 7, (1991) 489-499
21. ARToolKit, http://www.hitl.washington.edu/artoolkit/

GrayCut - Object Segmentation in IR-Images

Christian Ruwwe and Udo Zölzer

Department of Signal Processing and Communications
Helmut-Schmidt-University
University of the Federal Armed Forces
Hamburg, Germany
christian.ruwwe@hsu-hh.de

Abstract. Object segmentation is a crucial task for image analysis and has been studied widely in the past. Most segmentation algorithms rely on changes in contrast or on clustering the same colors only. Yet there seem to be no real one-and-for-all solution to the problem. Nevertheless graph-based energy minimization techniques have been proven to yield very good results in comparison to other techniques. They combine contrast and color information into an energy minimization criterion. We give a brief overview of two recently proposed techniques and present some enhancements to them. Furthermore a combination of them into the *GrayCut* algorithm leads to suitable results for segmenting objects in infrared images.

Keywords: Image Segmentation, Energie Minimization, Graph-based Techniques, Infrared Images.

1 Introduction

There has been a long road on the search to the holy grail of image segmentation [1], [2], [3], [4]. Most segmentation algorithms rely on changes in contrast or on grouping the same colors only. Yet there seem to be no real one-and-for-all solution to the problem.

Recent advances show graph-based techniques are superior in terms of quality and user interaction compared to other algorithms [5], [6], [7]. They combine contrast and color information into an energy minimization criterion. We will explain them in the following section in a little more detail, before we show how our now proposal fits into this evolution.

After the following short introduction into two graph-based algorithms, the next section contains the core description for our new segmentation algorithm, called *GrayCut*. Some experimental results and examples show the usage of our proposed method for segmentation of objects in gray-valued images - especially for infrared (IR) images. Finally, we conclude with a summary and give a brief outlook on future work.

1.1 GraphCut

The idea of using an energy minimization technique for image segmentation and solving it with graph-based algorithms was first described by Greig, Porteous and Seheult in 1989 [8]. We will describe the *GraphCut* as it was later refined by Boykov and Jolly [5].

It combines the two already known approaches for image segmentation: algorithms based on colors (or more precisely gray-levels) and segmentation based on the contrast in different regions of an image. For successful segmentation the energy formulation

$$E(z) = P(z) + \gamma \cdot C(z) \tag{1}$$

has to be minimized. The weighting parameter γ controls the importance of one term over the other.

The fidelity term $P(z)$ gives rise to a cost function, which penalizes false classification of a pixel z of the image I to the foreground $\alpha = 1$ or to the background $\alpha = 0$. Since the user provides a so-called trimap, where two regions - sure foreground and sure background - has to be defined, one can easily calculate a probability distribution and cost functions $p_{z,\alpha}$ from the gray-valued pixels and the image histograms of these two regions to be

$$P(z) = \sum_{z \in I} p_{z,\alpha} \ . \tag{2}$$

Costs can be calculated from the negative log-likelihood of the probability belonging either to the foreground or to the background.

Second a prior term $C(z)$ representing the pairwise interactions between neighboring pixels is calculated from the contrast between each two neighboring pixels z and $\hat{z}$

$$C(z) = \sum_{(z,\hat{z}) \in N} c_{z,\hat{z}} \ . \tag{3}$$

The neighborhood N is chosen such that only neighboring pixels around the segmentation boundary are summed up. These are only pixels z and $\hat{z}$ belonging to two different foreground/background maps: $\alpha_z \neq \alpha_{\hat{z}}$. Only a 4-way neighborhood is used here. Therefore, the minimization criterion is to find the shortest possible segmentation border that gives the smallest sum over its contrast terms.

The contrast between neighboring pixels z and $\hat{z}$ can be expressed as

$$c_{z,\hat{z}} = \exp\left(-\frac{(I_z - I_{\hat{z}})^2}{2\sigma^2}\right) \ , \tag{4}$$

where I_z is the gray-value of the pixel z in the range $0 \ldots 1$. The variance σ^2 over all differences in intensity can be seen as the noise floor present in the image. Choosing this parameter carefully lets the contrast term successfully switch between almost zero for high contrast and one vice versa. However, other functions, separating noise from real contrast in the same manner, are also possible.

From these two properties of each pixel - one belonging to the object or the background, the other being an edge or not - an undirected graph is built [9]. More precisely a so called S/T-graph is built, where the two terminals S and T represent the object respectively the background. Edges from and to these terminals are weighted with the corresponding foreground/background costs $p_{z,\alpha}$.

Neighboring pixels are connected with edges in 4-way neighborhood, weighted with the corresponding contrast terms $c_{z,\hat{z}}$.

Finally using a standard minimum-cut/maximum-flow (MC) algorithm has been proven to give the optimal segmentation border in terms of the energy formulation $E(z)$ defined in (1). The segmentation border corresponds to the edges representing the minimum cut in the graph.

1.2 GrabCut

GrabCut published in 2004 by Rother, Kolmogorov and Blake [6] extends this useful scheme to color images. Instead of gray-level histograms, it makes use of Gaussian mixture models (GMM). Background and foreground are each described with five full-covariance Gaussian components $M_{z,k}$. So the fidelity term $P(z)$ is now calculated from the superposition of the Gaussian components

$$M_{z,k} = \frac{1}{2\pi\sqrt{\Sigma_k}} \exp\left(-\frac{1}{2}(I_z - \mu_k)^T \Sigma_k^{-1}(I_z - \mu_k)\right) , \tag{5}$$

where the term I_z now reflects a three-valued RGB color of the pixel z. The μ_k are the mean color of each component and Σ_k are full-covariance matrices reflecting color dependencies between the three color layers. Adaptation of the probability distributions $M_{z,k}$ to the RGB colors is carried out with the iterative expectation maximization (EM) algorithm [10], according to a predefined trimap given by the user.

Due to the three-dimensional color space, the contrast $c_{z,\hat{z}}$ is now calculated as

$$c_{z,\hat{z}} = \exp\left(-\frac{||I_z - I_{\hat{z}}||^2}{2\sigma^2 \cdot ||z - \hat{z}||}\right) , \tag{6}$$

where the norm $||I_z - I_{\hat{z}}||$ is the Euclidian distance in RGB space and $||z - \hat{z}||$ indicates the spatial (Euclidean) distance between two neighboring pixels z and $\hat{z}$ (*GrabCut* uses a 8-way connectivity).

In this manner, the whole algorithm is laid out in an iterative way: after each EM iteration, an S/T-graph is built up like in the *GraphCut* and solved with the minimum-cut algorithm. The resulting segmentation border is used to update the trimap describing foreground and background regions. This new trimap is used for the next EM iteration and so on.

The alternating usage of EM steps and MC solutions guarantees the proper monotonic energy minimization over time. The amount of changes in the overall energy $E(z)$ between two iterations might be used as a suitable stopping criterion for the algorithm.

2 GrayCut for Infrared Images

Infrared images are gray-level representations of infrared emissions that cannot be seen directly by a human observer. The colorization is therefore purely virtual, almost like in x-ray images, where bright gray-values represent dense materials. Colorization is normally chosen such that warm (hot) regions are displayed in a bright gray-value, whereas cold regions are shown in darker gray-values, but this mapping can also be inverted.

Furthermore, the relative infrared emissions depend on weather and climate conditions. In our application of naval ship images, the objects itself - the ships - might be warmer (or brighter) than the surrounding water, or even colder (i.e. darker) than the rest of the image. So using predefined colors for image segmentation will not work at all.

Additionally infrared images in general are not that sharp in displaying objects than daylight images. In fact the noise floor present in the images is always much greater than in typical man-made pictures. Therefore, pure edge-based image segmentation will not be sufficient at all, too.

We already used image segmentation with a *GrabCut*-related algorithm to separate ships in marine images from their surrounding water [7]. Nevertheless, this segmentation was carried out on RGB images. Now the infrared (IR) image comes into play. Since they are only gray-values, we combine the advantages of both algorithms: *GraphCut* which is gray-level-based, and *GrabCut*, that uses an iterative optimization scheme.

2.1 Gaussian Mixture Models

As in *GrabCut* we use Gaussian mixture models again, but this time only to find distributions in the two gray-scale histograms - the one for the user-defined background and one for the (unknown) rest. The possible range of values is reduced from the three-dimensional space of RGB colors to the purely one-dimensional gray-scale histogram. So the covariance-matrix Σ reduces to the simple scalar variance σ^2

$$M_{z,k} = \frac{1}{2\pi\sigma_k} \exp\left(-\frac{(I_z - \mu_k)^2}{2\sigma_k^2}\right) , \tag{7}$$

Using five Gaussian components for each model gives too much fragmentation. We found that using two or three components each is more suitable for gray-value segmentation.

2.2 Iterations

Adaptation of the Gaussian mixture models is of course carried out again by expectation maximization, so the whole algorithm is of an iterative nature.

Starting with a random distribution for EM learning as in *GrabCut* is not a good starting point for the segmentation task. We apply the very first EM step before the whole algorithm starts. This guarantees a proper initial distribution of the mixture models, but also ensures the adaptation to changes in the trimap based on intermediate segmentation results.

Since the possible range of values and the total number of components has been reduced, the overall algorithm performance haven been slightly increased.

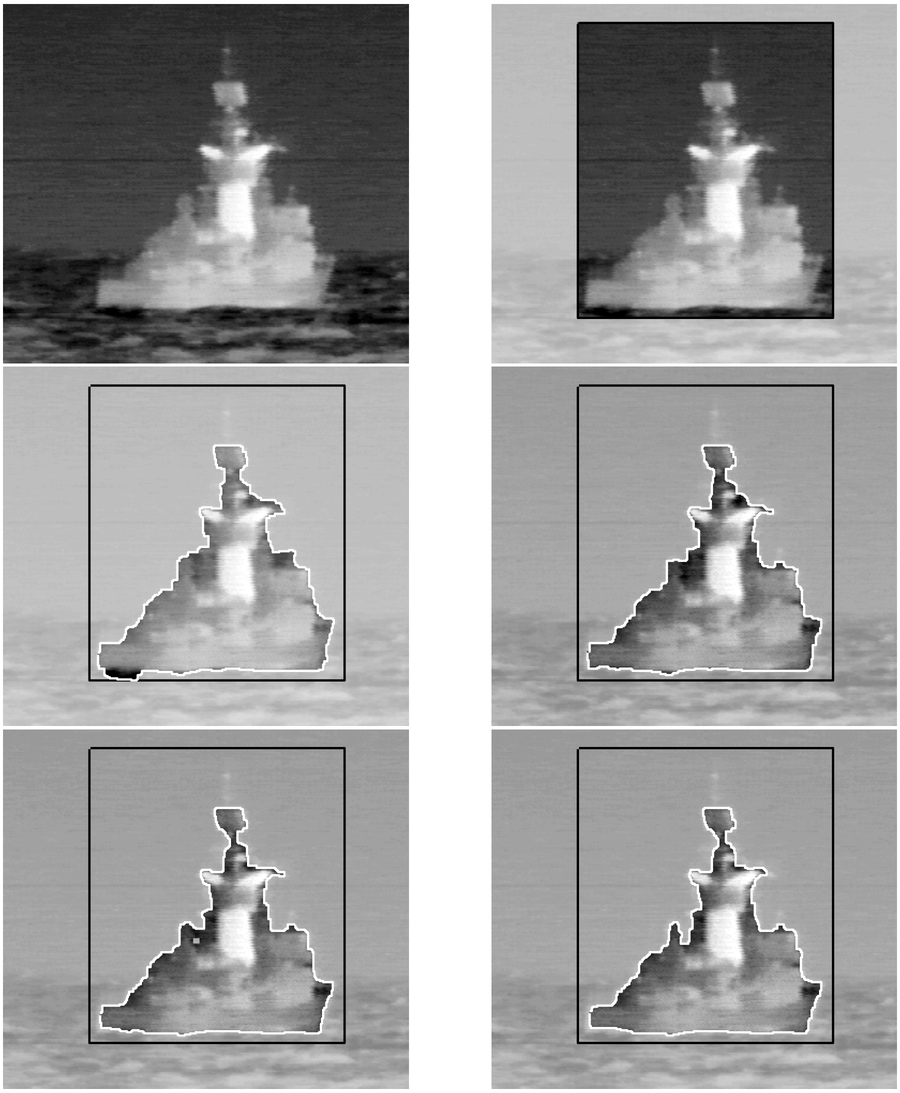


Fig. 1. An example infrared image and the segmentation results after the first five iterations: The first row shows the *input image* and the *background selection (black line)* as applied by the user. The next two rows show the *segmentation result (white line)* after the first four iterations of the *GrayCut* algorithm.

Moreover, less iterations are needed for the Gaussian components to adapt the gray-level histogram. Usually good results are already achieved after the first three to five iterations. Subsequent iterations only change few pixels directly at the segmentation border.

2.3 Post-processing

In same cases, the segmentation can be improved by applying additional post-processing operations between subsequent iteration steps. We have already shown in [7] that widening the calculated segmentation border with a morphological dilation operation gives superior results, when the color information in the image is too low for the fast adaptation by the EM algorithm.

We have successfully used the dilation operation with a disc-like structuring element. However, other structuring elements might be more usable to represent certain image content. For example, many vertical edges from mast and antennas present on many ships. This is scope of ongoing research.

Other morphological operations, like cleaning up small separated pixels in the foreground-map and leaving only the main object part, may be applied too in some cases. We found using these kind of post-processing techniques is more suitable when dealing with color images, than when segmenting gray-valued IR-images.

2.4 Don't-Care Map

When dealing with real-world applications, (infrared) images may contain several optical symbols laid over the original image. These lines and symbols are generated synthetically and therefore provide a very high contrast with sharp edges. Segmentation algorithm would normally react very strong on these sharp edges. Typically, this would lead to wrong segmentation results.

Since these synthetically added symbols always remain at the same position, we introduce an additional map called *don't-care*-map. This map distinguishes image regions with these special symbols from the rest of the real image and is used to exclude this pixel from the further algorithm.

These don't-care-map influences the construction of the S/T-graph in several ways: First sharp edges from this optical overlays don not carry any information for the segmentation task and should be rejected. Contrast terms $c_{z,\hat{z}}$ belonging to this edges are set to 1, simulating a homogeneous region with no edge at all.

Moreover the gray values of these pixels should not be adapted by the Gaussian mixture models, since it is not known whether the part of the image *behind* the optical symbol belongs to the object or to the background. Therefore pixels present the don't-care-map are not used during the EM-steps.

In addition to this uncertainness, any classification of pixels to the foreground or to the background has to be undone before the next minimum-cut can be solved. All pixels from the don't-care-map are set back to the *unknown*-state again, allowing the assignment of edge weights to the S/T-links again.

These slight modifications enable the algorithm to ignore special (predefined) parts of the image. The missing parts of the segmentation can move around freely inside the don't-care-regions, since all constraints from the contrast terms are cleared. Therefore solving the minimum-cut connects all valid parts of the segmentation with the shortest possible segmentation border through this unknown region.

2.5 User Interaction

We want to have as few user interactions as possible needed to carry out the segmentation task. Defining only a rough background region seems to satisfy our goal, but further improvements into fully automated segmentation are desirable.

Nevertheless, the iterative algorithm structure enables the user to redefine his trimap with additional background or additional foreground regions. Defining foreground in the beginning is usually not very helpful for the segmentation task, and slows down the whole process from the user point of view.

Only in the rare case of *difficult* images, the user has to give additional constraints to the algorithm and apply few more iterations to achieve the desired result.

3 Experimental Results and Discussion

Our experiments on infrared images showing navy ships demonstrate the usability of the proposed *GrayCut* algorithm.

Figure 1 shows an example segmentation on one of the infrared ship images. The first four iterations are shown. The black line indicates the originally drawn background box by the user and evolves over time to the segmentation border in white. The difference between each iteration step is getting lower and lower and more than five iterations are not needed at all.

Figure 2 demonstrates the evolution of the six Gaussian components during five iterations. In the beginning, all six distributions show an almost equal behavior due to the random initialization. In each iteration, the components tend to separate from each other by the expectation maximization. The dotted Gaussians represents the background, the solid lines are components of the foreground model. Again, only the first few iterations are usable. The last (fifth) iteration does not bring any real gain in the overall segmentation quality.

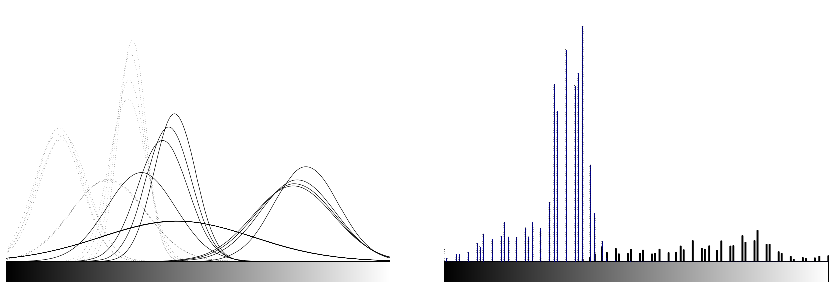


Fig. 2. Evolution of the *Gaussian mixture model components (left)* over five iterations for the *foreground (solid)* and for the *background (dotted)* and the real *gray-value histograms (right)* of the *foreground (solid)* and the *background (dotted)*

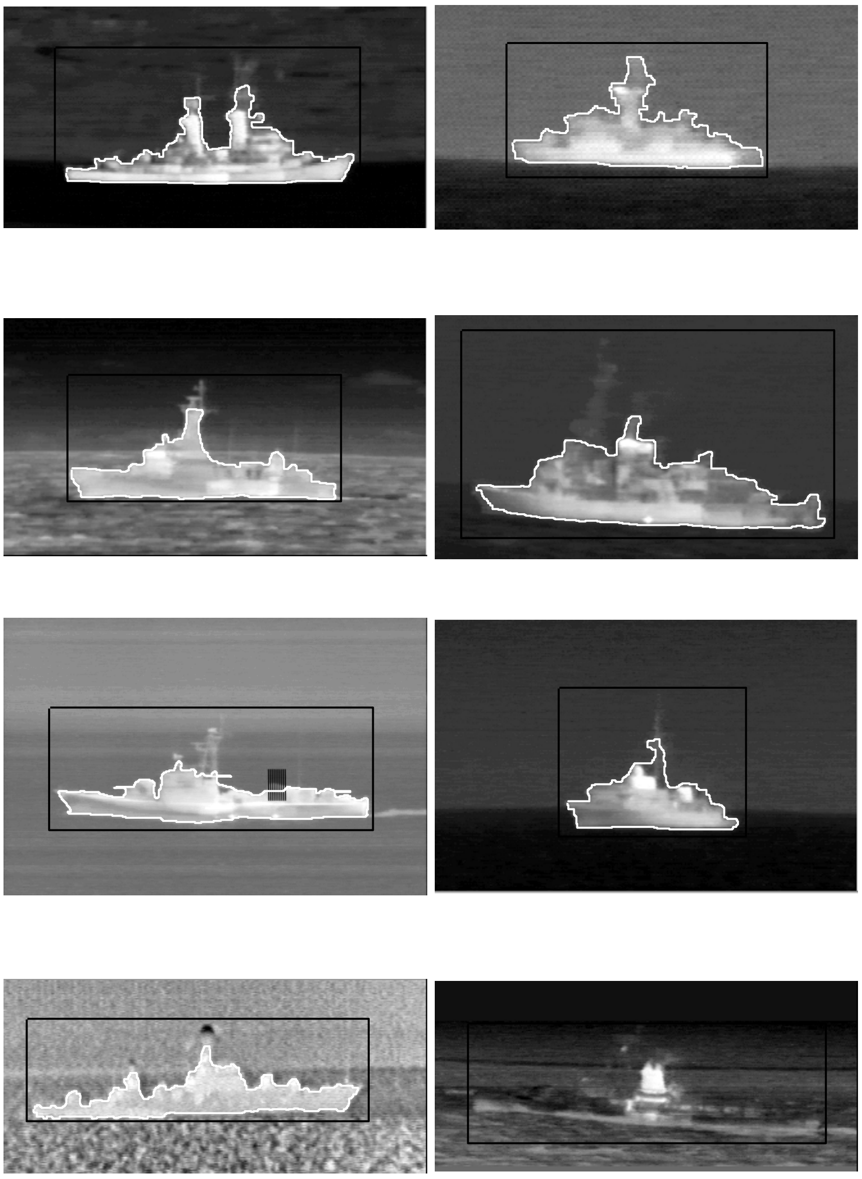


Fig. 3. More segmentation results on different images: The *black line* indicates the *background region* as defined by the user, the *white line* shows the *segmentation result* of the proposed *GrayCut*-algorithm after 10 iterations (or no result at all as in the last case in the lower right corner)

The right part of figure 2 shows the real histogram of the example image. Again, the dotted parts indicate the background and solid bars show the histogram of the foreground. It is clearly visible, that the amount of information present in the object is clearly less than in the rest of the image: the number of pixels belonging to the dark background region is much greater than the few ones corresponding to the foreground object. This is one reason for applying two different Gaussian mixture models describing foreground and background separately.

We used a set of 79 infrared images showing different levels of quality. Figure 3 shows some examples of the segmentation results. The black rectangle is the initial background selection by the user, whereas the white line shows the final segmentation result (after 10 iterations). The left image in the third row shows an example application for the *don't-care*-map where the black lines overlaid in the rear part of the ship are ignored by the segmentation border.

The proposed *GrayCut* algorithm is able to give 23 out of 79 suitable results without the need of additional user interaction. Additional 25 results can be improved with refinement of foreground and background regions by the user. The last 31 images result in wrong or even no segmentation borders at all. This is mainly due to the bad signal-to-noise ratio of the infrared images. We presume even a human user might have difficulties in drawing a suitable segmentation border in images of the last type seen in figure 3.

4 Conclusions and Future Work

We have used already known algorithms to derive a new method called *GrayCut* for segmenting in gray-level images. Some special extensions to the basic algorithm have been introduced, especially the *don't-care*-map to ignore some parts of the image. The development was mainly driven by infrared ship images, but we think this is also applicable in other fields if image processing and analysis, like in X-ray images for medical applications.

User interaction is reduced to the absolute minimum, i.e. only the background region is needed and has to be given by the user. The overall goal is to yield a perfect segmentation without additional user input, but the user can correct bad results by giving more constraints. Since the scene structure - a ship in the water - is nearly always the same, a fully automated algorithm is desirable, where no user input at all is needed.

The previous experiments show the usability of the proposed *GrayCut* algorithm. Nevertheless, different enhancements and parameter tuning have been shown to yield superior results. As a drawback, these parameters have to be adjusted carefully by the user before applying the segmentation and depend on the image content and the image quality. Currently no automatic parameter adjustment and post-processing selection is available. This is still scope of future research.

On the other hand, pre-processing steps might be useful to improve the image quality before applying the segmentation algorithm. Namely noise reduction as already proposed in [11] might be a very suitable step. An integration of these image enhancement algorithms has to be investigated in the future.

References

1. Mortensen, E., Barrett, W.: Intelligent scissors for image composition. ACM Siggraph (1995) pp. 191-198.
2. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. International Conference on Computer Vision (1987) pp. 259-268.
3. Friedland, G., Jantz, K., Rojas, R.: Siox: Simple interactive object extraction in still images. International Symposium on Multimedia (2005) pp. 253-259.
4. Wippig, D., Klauer, B., Zeidler, H.: Extraction of ship silhouettes using active contours from infrared images. International Conference on Computer Vision (2005) pp. 172-177.
5. Boykov, Y., Jolly, M.: Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images. International Conference on Computer Vision (2001) Vol. I, pp. 105-112.
6. Rother, C., Kolmogorov, V., Blake, A.: Grabcut - interactive foreground extraction using iterated graph cuts. Proc. ACM Siggraph (2004) pp. 309-314.
7. Rusch, O., Ruwwe, C., Zoelzer, U.: Image segmentation in naval ship images. 11. Workshop Farbbildverarbeitung (2005) pp. 63-70.
8. Greig, D., Porteous, B., Seheult, A.: Exact maximum a posteriori estimation for binary images. Journal of the Royal Statistical Society (1989) Series B, 51(2):271279.
9. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) (2004) Vol. 26, No. 2, pp. 147-159.
10. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society (1977) Series B, 39(1), pp. 138.
11. Wippig, D., Klauer, B., Zeidler, H.: Denoising of infrared images by wavelet thresholding. International Conference on Industrial Electronics, Technology & Automation (2005)

Unsupervised Clustering of Shapes

Mohammad Reza Daliri and Vincent Torre

SISSA, Via Beirut 2-4, 34014 Trieste, Italy
`Daliri@sissa.it, Torre@sissa.it`

Abstract. A new method for unsupervised clustering of shapes is here proposed. This method is based on two steps: in the first step a preliminary clusterization is obtained by considering the distance among shapes after alignment with procrustes analysis [1],[2]. This step is based on the minimization of the functional $\theta(N_{cluster}) = \alpha N_{cluster} + (1/N_{cluster}) dist(c_i)$ where $N_{cluster}$ is the total number of clusters, $dist(c_i)$ is the intra-cluster variability and α is an appropriate constant. In the second step, the curvature of shapes belonging to clusters obtained in the first step is examined to i) identify possible outliers and to ii) introduce a further refinement of clusters. The proposed method was tested on the Kimia, Surrey and MPEG7 shape databases and was able to obtain correct clusters, corresponding to perceptually homogeneous object categories. The proposed method was able to distinguish shapes with subtle differences, such as birds with one or two feet and to distinguish among very similar animal species. ...

1 Introduction

Computer vision aims at building machines able to recognize and categorize objects as the human brain does [3]. Recognition and categorization should be fast and efficient also with very cluttered and complex images. Finding a solution to this problem is rather difficult for several reasons. Firstly, the required level of categorization is not obvious: do we want to distinguish a bird from a bat? Or are we aiming at recognizing a juvenile finch from a male adult finch? Based on the research carried out by some cognitive scientists [4], categorization is performed at several levels. Secondly, there is a natural variability within different categories and the expected variability does not necessarily match the real one. Thirdly, characterization should be invariant to rotation, scale and, translation, to certain deformations and most important to partial occlusions.

Objects have several properties that can be used for recognition, like shape, color, texture, brightness. Each of these cues can be used for classifying objects. Biederman [5] suggested that edge-based representations mediate real-time object recognition. In his view, surface characteristics such as color and texture can be used for defining edges and can provide cues for visual search, but they play only a secondary role in the real-time recognition. There are two major approaches for shape-based object recognition: 1) boundary-based, that uses contour information [6], [7], [8], [9], [10], [11] and 2) holistic-based representation, requiring more general information about the shape [12], [13]. Categorization is often obtain by clustering [14], [15], [16]. Although all these methods, provides very reasonable results, they do not have an automatic way to find the number of clusters.

In this manuscript we address the issue of unsupervised clusterization of shapes by using the contour information in a global and local way. Our procedure consists of a first preprocessing of shapes which are aligned pairwise and scaled using the procrustes shape analysis [1], [2]. This preprocessing provides the requested invariance for scale, rotation, symmetric mirroring or flipping. Shapes are then clustered by minimizing the functional (9), based on the global computation of the distance between contour shapes. Subsequently, the curvature is analyzed for the detection of possible outliers and for the identification of further clusters based on local differences in the curvature.

2 Initial Pre-processing of Shapes

Let $(xi, yi)_{S_k}$ $i = 1, ..., M$ be the contour of the shape S_k with k=1,...,N, where N is the total number of shapes to be categorized or, more formally, clusterized. Shape contours are interpolated so that they have a common number (m) of points (such as 100 points per contour) and contours are aligned in pairs by using procrustes analysis [1]. Given two contours S_1 and S_2 with the same number of points N, the Procrustes analysis finds the best transformation T of one contour, i.e.:

$$S_2{}^* = T(S_2), \tag{1}$$

such that the distance between the two contours S_1 and $S_2{}^*$ is minimum [17]:

$$\|S_2{}^* - S_1\|^2. \tag{2}$$

Restricting transformation to the similarity case, we have:

$$T\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a & -b \\ b & a \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}. \tag{3}$$

If the two shapes have the same center of mass located in the origin, the optimal transformation is:

$$((t_x, t_y) = (0, 0)), \tag{4}$$

$$a = (S_1 S_2)/\|S_1\|^2, \tag{5}$$

$$b = (\sum (x_i y_i\prime - y_i x_i\prime)/\|S_1\|^2), \tag{6}$$

where $S_1 = (x_i, y_i)$ and $S_2 = (x_i\prime, y_i\prime)$. Since we do not have the optimal correspondence for the two point sets, it is necessary to consider N possible cyclic renumbering for each shape, so the optimal alignment will be obtained when the full Procrustes distance is minimal regarding these renumbering. Given two aligned shapes S_k and S_h, the distance between them, $d(S_k, S_h)$ is defined as the

χ^2 test between histograms of point distributions computed by shape contexts [10]:

$$d(S_k, S_h) = \frac{1}{2} \sum_{i=1}^{mI} \frac{(h_k(i) - h_h(i))^2}{h_k(i) + h_h(i)}, \tag{7}$$

where $h_k(i)$ and $h_h(i)$ denote the I-bin normalized histograms of pairs of aligned points and m the number of points over the contour. In this way one obtains all pairwise distances $d(S_k, S_h)$ $k, h = 1, \ldots, N$. An other possible way to compute the pair distances is by using the Euclidean distance:

$$d(S_k, S_h) = \sqrt{\sum_{i=1}^{m} (x_{ki} - x_{hi})^2 + (y_{ki} - y_{hi})^2}, \tag{8}$$

where (x_{ki}, y_{ki}) and (x_{hi}, y_{hi}) are the aligned contour points. Shape context usually gives more precise results, because it represents better the general properties of shapes. For each aligned contour S_k we compute its curvature $\kappa_k(i)$ $i = 1, \ldots, m$ by using a B-spline interpolation as described in [18].

3 First Clustering Based on Intra-shape Distance

Let us consider N shapes, which we want to cluster in different categories. The main idea is to minimize the functional:

$$\theta(N_{cluster}) = \alpha N_{cluster} + \frac{1}{N_{cluster}} \sum_{1}^{N_{cluster}} dist(c_i), \tag{9}$$

where $N_{cluster}$ is the number of clusters, $dist(c_i)$ is the intra-cluster distance, i.e. the scaled average squared distance between shapes in the cluster c_i and α is a parameter controlling the grain of the clustering. For large values of α the $N_{cluster}$ will be small and viceversa for small values of α, $N_{cluster}$ approaches the value of N. The intra-cluster distance $dist(c_i)$ is defined as:

$$dist(c_i) = \frac{2}{N_{ci}} \sum_{S_k, S_h \in c_i, k<h} d(S_k, S_h)^2. \tag{10}$$

Intuitively the correct number of clusters N_c is found when the addition of an other cluster does not reduce significantly the average intra-cluster distance $avdis(N_c)$:

$$\frac{1}{N_{cluster}} \sum_{1}^{N_{cluster}} dist(c_i), \tag{11}$$

i.e. the correct number of clusters $N_{cluster}$ is found when $\theta(N_{cluster} + 1) - \theta(N_{cluster})$ is very small. This condition implies that α must be close to:

$$\frac{avdis^*}{(N_{cluster}{}^* + 1)}, \tag{12}$$

where $avdis^*$ is the desired intracluster average distance and ${N_{cluster}}^*$ the expected number of clusters. Minimization of the functional (9) is obtained by a progressive creation of new clusters. Initially two clusters with approximately the same number of shapes are created by a random selection. Having the initial clustering functional (9) is minimized by a gradient descent type algorithm, in which, at every iteration, the optimal clusterization of each shape is considered. This procedure continues until the value of (9) became stable for the complete cycle in which all shapes S_j are considered. Then a new cluster is created and the entire procedure is repeated.

4 Second Clustering Using Distance Between Curvatures

As we have observed during our experimentation the unsupervised clustering described in section 3 does not always provide a categorization which appears to be perceptually correct. Indeed, in a cluster of fish for instance, other animals may often be present because the similarity measure corresponding to (7) or (8) is not adequate or the minimum found during the minimization of (9) is simply a local minima. Therefore a second step, based on an other similarity measure is necessary to find outliers and to refine the obtained categorization/clusterization. This second step is based on the analysis of the curvature $\kappa_k(i)$ of shapes. For each cluster C_m - in which shapes have been aligned as described in [4] - we compute the average curvature ${\kappa_m}^*(i)$ and its standard deviation $s.d.{\kappa_m}^*(i)$. For each shape S_{jm} within cluster C_m , we compute:

$$d({\kappa_{jm}}^*, {\kappa_m}^*) = \sum_i ({\kappa_{jm}}^*, {\kappa_m}^*)^2. \tag{13}$$

If $d({\kappa_{jm}}^*, {\kappa_m}^*)$ is larger than a threshold the shape S_{jm} is considered an outlier and is either moved to an other cluster or a new cluster is created.

5 Experimental Results

5.1 Estimating the Parameter of α with Kimia Database

A subset of 25 different shapes of Kimia database (see Fig.1A) from 6 different classes was used as ground truth to estimate the parameter α of functional (9). When α was set equal to 0 - as expected - θ decreased with $N_{cluster}$ (Fig.1B). When the value of α was $3.1714 * 10^6$ - with the shape context distances - θ had a minimum for $N_{cluster}$ equal to 6, as required (Fig.1C). Therefore a good estimate for $avdis^*$ is $2.9 * 10^7$.

5.2 Comparison with a Small Example

In [15], a small number of shapes consists of 25 shapes from the Surrey fish database [19] has been used for clustering. The proposed method is based on pairwise similarity clustering like ours, but for the pairwise similarity measure

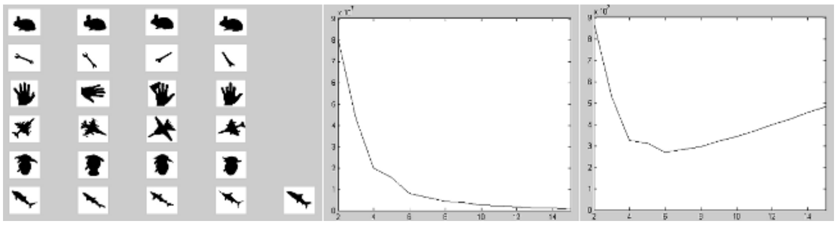


Fig. 1. A) Clustering of Kimia database with the algorithm described in the text. B) The energy function $\theta(N_{cluster})$ defined in (9) with α equal to 0. and C) The energy function $\theta(N_{cluster})$ with α equal to $3.1714 * 10^6$.

they use geodesic path between two shapes, which is the path that uses minimum energy to bend one shape into the other. For their method, they need to fix the number of clusters or they can obtain hierarchial clustering. In this small example they set the number of clusters to 9. Fig.3 shows the result of clustering proposed in [15]. In Fig.4 we show the result of applying our method by fixing the number of cluster to 9, and Fig.5 presents the result of the unsupervised clustering using the estimated parameters from the previous part (The optimal number of clusters has obtained to be 7 with the algorithm). For all three Figures, each column represents a cluster.

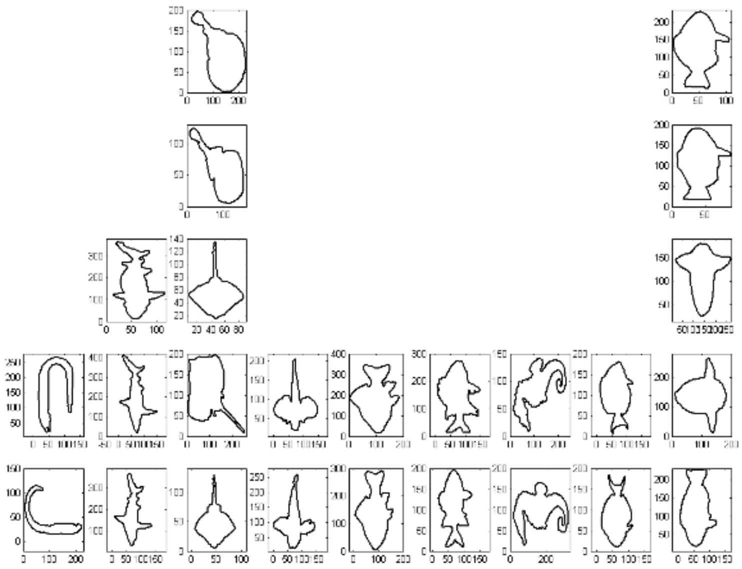


Fig. 2. A clustering of 25 fish shapes into 9 clusters reproduced from [15]

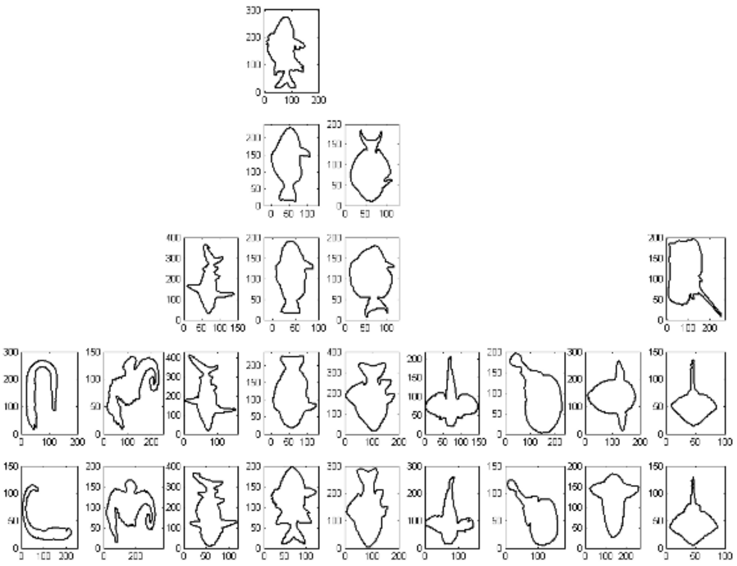


Fig. 3. A clustering of 25 fish shapes with the proposed method in this paper having fixed the number of clusters to 9

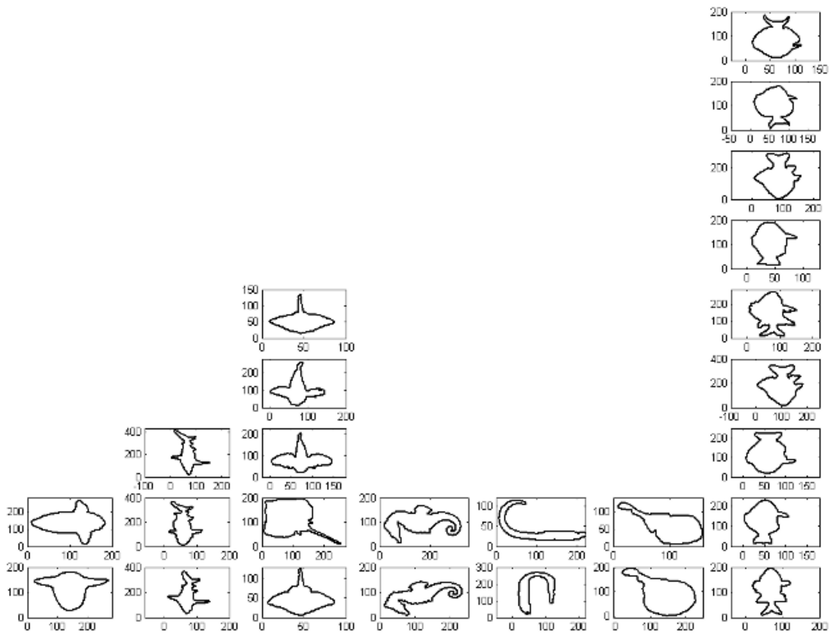


Fig. 4. A clustering of 25 fish shapes with the unsupervised clustering method proposed in this paper. The optimal number of clusters has obtained to be 7 with the algorithm.

5.3 Removing the Outliers with the Second Clustering

Fig.5A illustrates a cluster obtained from the MPEG7 Database of cows, birds, camels, flat-fish and fish. Inside the cluster corresponding to the flat-fish one camel is present. As shown in Fig.5B after alignment, the shape of this outlier camel is as close to a flat-fish as two different flat-fish are close to each other (see Fig.5C and D). By analyzing the curvature of the shapes of the cluster of fig.5A, the outlier is easily detected (see Fig.5E).

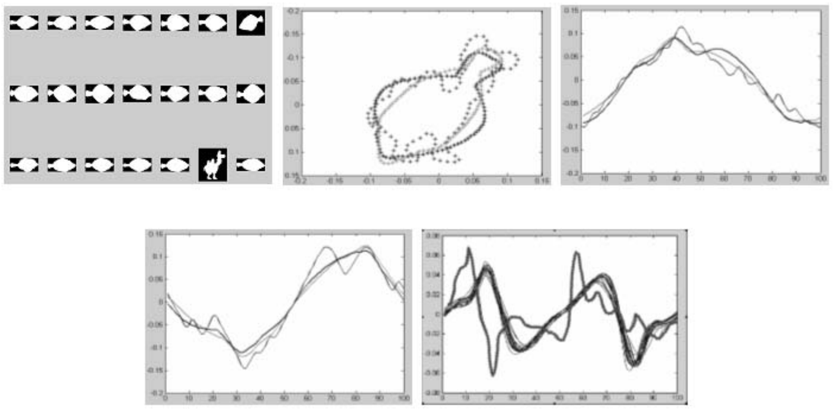


Fig. 5. A) Cluster of shapes corresponding to different flatfish obtained by using the first clustering algorithm B) Alignment of the Camel with two flatfish C and D) the x and y component respectively of the contours shown in B. The same colour was used for the same shape. E) Alignment of curvatures identify the outlier, i.e. the camel.

5.4 Some Result with MPEG7 Shape Database

400 different shapes from 20 different classes in the MPEG7 shape database were clustered according to the proposed procedure. Some of the clusters obtained by the first clustering step are shown in Fig.6. In red are indicated the outliers in the different identified clusters. These outliers were easily identified by analyzing the curvature described in section 4.

6 Discussion

The proposed method for clustering shapes is based on two steps: in the first step a preliminary clusterization is obtained by minimizing the functional (9). In the second step, the curvature of shapes is examined in order to identify possible outliers. This method was tested successfully on the Kimia, Surrey and MPEG7 shape databases. The suggested clusterization in conjunction with image segmentation [20] maybe also suitable for recognizing shapes in cluttered real images.

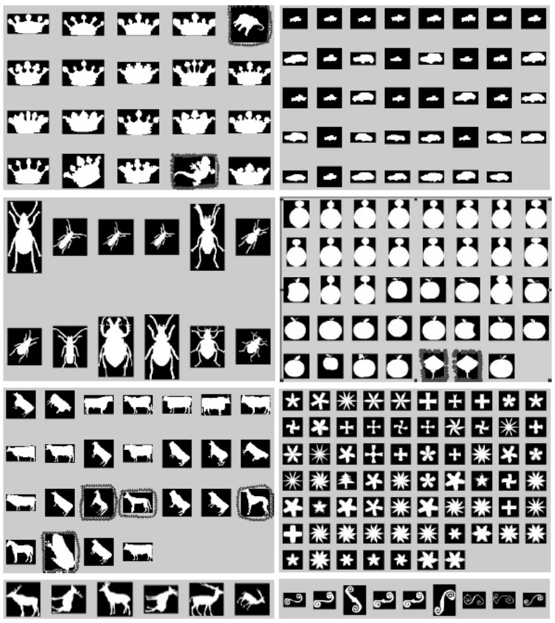


Fig. 6. Clusters from the MPEG7 database. In red are indicated the outliers detected by the analysis of the curvature described in the text.

References

1. Bookstein, F.: Landmark methods for forms without landmarks: Localizing group differences in outline shape. Medical Image Analysis **1** (1997) 225–244
2. Goodall, C.: Procrustes methods in the statistical analysis of shape. J. Royal Statistical Society, Series B **53** (1991) 285–339
3. Ullman, S., ed.: High-level Vision. MIT press (1996)
4. Edelman, S., ed.: Representation and Recognition in Vision. MIT press (1999)
5. Biederman, I., Ju, G.: Surface versus edge-based determinants of visual recognitions. Cognitive Psychology **20** (1988) 38–64
6. Coggins, J.: A statistical approach to multiscale, medial vision. Technical report, Department of Computer Science, University of North Carolina at Chapel Hill (1992)
7. Blum, H.: A transformation for extracting new descriptors of shape. In Wathen-Dunn, W., ed.: Models for the Perception of Speech and Visual Form, Cambridge, USA, MIT Press (1967) 362–380
8. Sebastian, T., Klien, P., Kimia, B.: Recognition of shapes by editing their shock graphs. IEEE Transaction on PAMI **26** (2004) 550–571
9. Mokhtarian, F., Mackworth, A.: Scale-based description and recognition of planar curves and two-dimensional shapes. IEEE Transaction on PAMI **8** (1986) 34–43
10. Belongie, S., Malik, J.: Shape matching and object recognition using shape contexts. IEEE Transaction on PAMI **24** (2002) 509–522

11. Arbter, K., Snyder, W., Burhardt, H., Hirzinger, G.: Application of affine-invariant fourier descriptors to recognition of 3-d objects. IEEE Transaction on PAMI **12** (1990) 640–647
12. Rivlin, E., Weiss, I.: Local invariants for recognition. IEEE Transaction on PAMI **17** (1995) 226–238
13. Murase, H., Nayar, S.: Visual learning and recognition of 3-d objects from appearance. Int. J. Computer Vision **14** (1995) 5–24
14. Duta, N., Sonka, M., Jain, A.: Learning shape models from examples using automatic shape clustering and procrustes analysis. Lecture Notes in Computer Science (1999)
15. Srivastava, A., Joshi, S., Mio, W., Liu, X.: Statistical shape analysis: Clustering, learning, and testing. IEEE Transaction on PAMI **27** (2005) 590–602
16. Jain, A., Murty, M., Flynn, P.: Data clustering: A review. ACM Computing Surveys **31** (1999) 264–323
17. Cootes, T., Taylor, C., Cooper, D., Graham, J.: Active shape modelstheir training and application. Comput. Vis. and Imag. Underst. **61** (1995) 38–59
18. Medioni, G., Yasumoto, Y.: Corner detection and curve representation using cubic b-splines. Computer Vision, Graphics and Image Processing **39** (1987) 267–278
19. Mokhtarian, F., Abbasi, S., Kittler, J.: Efficient and robust shape retrieval by shape content through curvature scale space. In: Proc. First Int'l Conf. Image Database and Multi-Search. (1996) 35–42
20. Vanzella, W., Torre, V.: A versatile segmentation procedure. IEEE Transactions on SMC, Part B **36** (2006) 366–378

Markerless Pose Tracking for Augmented Reality

Chunrong Yuan

Fraunhofer Institute for Applied Information Technology
Schloss Birlinghoven, 53754 Sankt Augustin, Germany
chunrong.yuan@fit.fraunhofer.de

Abstract. In this paper a new approach is presented for markerless pose tracking in augmented reality. Using a tracking by detection approach, we estimate the 3D camera pose by detecting natural feature points in each input frame and building correspondences between 2D feature points. Instead of modeling the 3D environment, which is changing constantly and dynamically, we use a virtual square to define a 3D reference coordinate system. Camera pose can hence be estimated relative to it and the calculated 3D pose parameters can be used to render virtual objects into the real world. We propose and implement several strategies for robust matching, pose estimation and refinement. Experimental evaluation has shown that the approach is capable of online pose tracking and augmentation.

1 Introduction

The main concept behind augmented reality (AR) is to integrate extra perceptible elements such as sound, graphics, image, video, force feedback, etc., into a user's real–world environment, for the purpose of improved understanding and interaction. Since vision plays an important role in human perception, most AR applications need to render computer generated graphics into a user's field of view. One option for allowing visual perception of the virtual objects is to let the users wear head–mounted displays (HMD). Through the HMD, users can see both the virtual and real world, with the virtual world aligned seamlessly to the real environment.

AR requires accurate registration and visualization of virtual objects in 3D. In order to render a virtual object into the real world, a virtual camera has to be placed in the same position and orientation as the real camera. In other words, at the location through which a person observes the surroundings. This results in a problem of the 3D localization of the HMD.

With the improvement of computer vision algorithms and the emergence of low–cost optical sensors, mounting a camera on the HMD has become common practice. Once the 6–DOF (degree of freedom) pose of the camera has been estimated, the pose of the HMD as well as that of the virtual camera can be computed easily.

In AR, pose tracking of the head–mounted camera is quite a challenging task. The motion of the camera is very dynamic and quite unpredictable. The 3D camera pose must be estimated online for each video frame. The computational complexity of the pose estimation algorithm should be reasonable, and the robustness as well as accuracy has to be ensured. In this work, we contribute to the state of the art by introducing a robust markerless tracking approach which can support accurate online augmentation in unconstrained environments.

2 Previous Research

In the computer vision community, much research has been carried out on issues relating to tracking and pose estimation. One approach is based on structure from motion [1]. Some authors [2] [3] [4] try to solve simultaneously both structure and motion by using a whole image sequence that has been captured beforehand. While capable of modeling the environment and estimating camera parameters, the complexity of the algorithms and the offline nature made them unsuitable for real–time AR applications. Optical flow and template matching based methods [5] [6] [7] use as a measure the sum of squared difference (SSD) for the estimation of the motion of an image template. In common with the probabilistic or prediction based approaches [8] [9] [10], camera motion is parameterized using models with low–order dynamics and the tracking systems can only handle situations where the frame–to–frame motion is regular and predictable. While real–time tracking can be achieved, these approaches work well only when changes of viewpoint do not occur rapidly. Another drawback is that the tracking system must be initialized properly and reinitialization has to be carried out frequently due to the drift problem. In contrast to the frame–to–frame tracking approach, most of the present camera-based tracking systems in AR apply a tracking by detection approach, where some reference objects are detected in each input frame and camera pose is estimated based on the known 3D configuration of these reference objects. The reference objects are called markers since they are designed and added manually to the tracking environment for the special purpose of easy detection and registration.

In the past, different markers have been designed for AR applications using either circular or planar markers. Interested readers may refer to [11] for some examples of marker–based tracking. One popular and also publicly available marker–based tracking software is the ARToolkit developed by [12]. Similar markers are used in [13], where a digital coding scheme is proposed for marker detection and recognition.

Based on image segmentation, marker–based tracking system is able to perform fast detection and identification of the markers. As a result, it can achieve real–time performance. However, one of the inherent drawbacks of such a system lies in the fact that marker detection is very sensitive to occlusion. Tracking stops immediately even if only a tiny portion of a marker is occluded. Although the use of multiple markers can provide a workaround, the occlusion problem remains unsolved.

An alternative to markers and image segmentation is the use of local features. In the field of object recognition, much research has been conducted into feature–based object descriptions and several interest point detectors aiming at reliable feature detection and matching have been proposed [14] [15] [16] [17] [18]. Recently, a comparative study has been carried out by [19], and the scale invariant feature transform, also known as SIFT [16], has been identified as one of the best feature detectors.

SIFT is relatively invariant to illumination and viewpoint changes, and is a good candidate for developing model–based tracking. In [20], SIFT features are used for establishing point correspondences between the input frame and those lying on a model which has to be built offline. The recent work of [21] also uses SIFT for tracking industrial parts in small environments. In common with [20], a CAD model of the scene has to be built beforehand.

There are approximately three strategies which have been used in developmenting markerless camera tracking. The first one uses localized features in combination with a 3D model of the environment [20] [21] [22] [23]. The second one requires manual initialization, and tracking is based on frame–to–frame differences [24] [25] [26]. The third one applies a strategy of sensor fusion and may use the first one for (re–) initialization and the second one for temporal registration [27] [28]. At the time of writing, the performance of current markerless tracking approaches is still beyond that of the marker based ones.

3 Approach and Technical Contribution

We built our tracking system by using the approach of tracking by detection. The system does not suffer from the drift problem, as pose estimation is carried out without the use of past frames. Moreover, it does not place assumption or constraint on camera motion. With a single off–the–shelf camera, no additional sensor is needed for the 6–DOF pose tracking. Unlike some of the previously mentioned works where manual initialization is needed at the beginning of the tracking process, our tracking system works full automatically.

In common with [20], we use SIFT for the detection of localized features from the natural environment. But our feature descriptor is more compact. Once local features are detected, we match them against each others based on a distance measure built upon the descriptors. To facilitate robust matching, outliers are deleted based on different criterion and by applying several robust algorithms. Based on the inliers, the initial pose of the camera is calculated, and is subsequently refined with a more advanced algorithm.

As indicated, most of current markerless tracking approaches require a 3D model of the environment for matching 2D features to those lying on the model. In addition to the complexity of building a model, such a strategy would result in performance problems when the model is very complex or the environment is dynamic. In contrast, our approach does not need to perform 3D engineering of the environment. Also since we do not need a 3D model, the matching of feature points is carried out completely in 2D, which increases the matching speed and simplifies the offline preparation.

For the robust tracking of the camera pose, we developed a new markerless approach that combines information from both the real and virtual world. We use a simple virtual model, a virtual square [1] with known size, to define a reference coordinate system. The virtual square can be embedded anywhere in the real world. This makes it possible to specify the 3D camera pose $\boldsymbol{P} = [\boldsymbol{R}|\boldsymbol{t}]$ relative to the reference coordinate system.

With the help of the virtual square, the 3D camera pose can be solved using a model–based approach. The model is not a 3D model of the environment, but a virtual model with simple configuration that is detectable from the real world.

During the offline stage, some reference frames can be captured, and the 2D position of the four corner points of the virtual square in the reference images can be determined. During the online tracking stage, once the 2D point correspondences are established

[1] We would like to point out that the "virtual square" used for registration purpose does not need to be a real square, it can also be a rectangle whose two sides have different length.

between the current camera image and one of the reference images, a 2D transform can be applied to locate the virtual square in the current camera frame. Based on the calculated 2D position as well as the 3D configuration of the virtual square, camera pose $\boldsymbol{P}$ can be calculated reliably. Once the 3×4 transform matrix $\boldsymbol{P}$ is computed, it can then be used for rendering virtual objects into the real world for augmentation purposes.

To our knowledge, the idea of using natural features together with a virtual model for 6–DOF pose tracking of a dynamic camera has not been explored before. These and the other features presented in this section constitute our main contributions.

4 Feature Detection and Description

Detection of salient feature points is done by using the original SIFT detector. The main purpose is to use DoG (difference of Gaussian) to detect and localize salient structures in the Laplacian scale space. A set of different scale levels are established by recursive filtering with variable Gaussian kernels, resulting in a set of DoG images $\{\boldsymbol{D}(x, y, \sigma)\}$.

On each $D(x, y, \sigma)$, local maxima/minima are sought at both the current and adjacent scales in order to identify feature points. Once feature location is found, a detail fit is carried out using a 3D quadratic function based on Taylor expansion. Through such a fitting process, it is possible to reject unstable extrema with a low contrast. At the same time, it allows a sub–pixel localization of the feature points. A further operation is carried out to eliminate those points that lie along edges. A principal curvature of $\boldsymbol{D}(x, y, \sigma)$ at the feature location is determined based on a ratio of the two eigenvalues of a 2×2 Hessian matrix $\boldsymbol{H}$. The final set of feature points are obtained by keeping only those points that have a small ratio between the two eigenvalues.

After the feature points are located, the next step is to build a descriptor to measure the distance between different features. In order to achieve a scale invariant description, it is important to record not only the location but also the scale σ at which a feature point is found. A further possibility is to have the descriptor calculated in a way which is relative to its orientation [16].

The calculation of the orientation of a feature point is based on the changes of intensity occurring in a neighborhood around the point in the Laplacian image $\boldsymbol{L}(x, y, \sigma)$. Suppose the input image is $\boldsymbol{F}(x, y)$, the Gaussian kernel is

$$\boldsymbol{G}(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\{-\frac{x^2 + y^2}{2\sigma^2}\}, \tag{1}$$

then the Laplacian image can be expressed as

$$\boldsymbol{L}(x, y, \sigma) = G(x, y, \sigma) * \boldsymbol{F}(x, y) \, . \tag{2}$$

From $\boldsymbol{L}(x, y, \sigma)$, a magnitude image $\boldsymbol{M}(x, y, \sigma)$ and an orientation image $\boldsymbol{\Theta}(x, y, \sigma)$ can be calculated as

$$\boldsymbol{M}(x, y, \sigma) = |\boldsymbol{L}(x+1, y, \sigma) - \boldsymbol{L}(x-1, y, \sigma)| + |\boldsymbol{L}(x, y+1, \sigma) - \boldsymbol{L}(x, y-1, \sigma)| \tag{3}$$

$$\boldsymbol{\Theta}(x, y, \sigma) = \arctan\{\frac{\boldsymbol{L}(x, y+1, \sigma) - \boldsymbol{L}(x, y-1, \sigma)}{\boldsymbol{L}(x+1, y, \sigma) - \boldsymbol{L}(x-1, y, \sigma)}\} \tag{4}$$

For each point detected, an orientation histogram with 16 bins is formed within a neighborhood around the point. The highest peak of the orientation histogram gives the dominant direction and is regarded as the orientation of the feature point. While the method in [16] allows for multiple feature points created at the same location and scale but with different orientations, our method provides a unique feature point whose orientation is dominant.

With the position, scale and orientation of each feature points identified, a rotation invariant descriptor can be built based on the $\boldsymbol{M}(x, y, \sigma)$ and $\boldsymbol{\Theta}(x, y, \sigma)$ images. We sample a 8×8 region around each feature point. The region is divided into four 4×4 subregions. On each of these subregions, an orientation histogram with 8 bins is built, resulting in a vector whose dimension is equal to 8. The feature point can hence be represented with a $4 \times 8 = 32$ dimensional vector, which is normalized in a subsequent step to a descriptor of unit length to make it invariant to affine changes in illumination.

5 Matching and Pose Estimation

The term matching refers to the matching of feature points to each other and matching of the current frame with those reference frames $\boldsymbol{f}_i, i = 1, \ldots, N$. Matching of two feature points is based on the Euclidean distance between the pair of descriptors. Candidates of matched pairs are selected based on a global threshold as well as the ratio between the closest distance and the second closest distance.

The same principle is applied to the matching of two frames. For each of the reference frames in the database, we calculate two measures. One is the total number of matched points n_i $(i = 1, \ldots, N)$ between a reference and the current frame. The other is the average distance of matched points between the two frames, which can be denoted as d_i $(i = 1, \ldots, N)$.

Usually a unique reference frame $\boldsymbol{f}_\kappa$ can be identified by finding the best match that has the biggest number of matched points as well as the smallest value of the average distance. This means, if $\kappa_1 = \text{argmax } \{n_i\}$, $\kappa_2 = \text{argmin } \{d_i\}$, then $\kappa = \kappa_1 = \kappa_2$.

If $\kappa_1 \neq \kappa_2$, then we just keep both reference frames $\boldsymbol{f}_{\kappa_1}$ and $\boldsymbol{f}_{\kappa_2}$ as candidates. Generally speaking, the frame $\boldsymbol{f}_{\kappa 1}$ with more matched pairs is the better candidate. But due to outliers, it could happen that $\boldsymbol{f}_{\kappa_1}$ is not the correct one. For this reason, we keep $\boldsymbol{f}_{\kappa_2}$ as another candidate and wait until next step for a single solution.

The matched pairs of feature points belonging to the current and the candidate reference frames are further verified by fitting an affine transform to those matched points. An affine transform $\boldsymbol{A}$ has six unknowns and can be solved by using three point correspondences. We use RANSAC [29] to initialize the transform matrix $\boldsymbol{A}$ based on randomly selected three point pairs. After initialization, we use a robust measure based on linear least squares to find the best affine transform between the current and a reference frame. Meanwhile, outlines are identified and only those points that agree with the transform are kept. Based on these inliers, a refined matrix $\boldsymbol{A}$ and an average deviation ϵ are calculated, which make it possible to find the best matched frame $\boldsymbol{f}_\kappa$ as well as the best transform matrix $\boldsymbol{A}_\kappa$ by keeping the candidate with the smaller ϵ.

As already mentioned in Section 3, we use a virtual square with known size for the purpose of pose estimation. Based on the plane on which the virtual square lies, we can define a world coordinate system whose origin is the center of the square. Having this as reference coordinate system, the 3D position of the four corner points $\boldsymbol{X}_j$ ($j = 1, \ldots, 4$) of the virtual square can be identified. From the set of reference frames, the 2D position of the four corners in each of them, i.e., $\boldsymbol{x}^i_j$ ($i = 1, \ldots, N, j = 1, \ldots, 4$), can also be determined.

During the on online tracking stage, suppose the current image has been found to be matched to the frame $\boldsymbol{f}_\kappa$ and the 2D affine transform between the two frames is $\boldsymbol{A}_\kappa$. This transform can be applied to the virtual square to find its 2D position in the current frames as $\hat{\boldsymbol{x}}_j = \boldsymbol{A}_\kappa \boldsymbol{x}^\kappa_j$. Now the problem becomes finding the 3D camera pose with four coplanar points whose 2D image coordinates and 3D world coordinates are known.

3D pose estimation based on 2D coplanar points is a special case of perspective–n–point (PnP) problem, which has been studied by [30] [31] [32]. For the estimation of the initial camera pose, we use the coplanar version of the POSIT (pose with iteration) algorithm [32]. Pose estimation based on a single projective image may suffer from the problem of pose ambiguity, resulting in the the initial calculated pose being inaccurate. In order to refine the initial pose, we use the robust estimation method proposed in [33] for pose refinement.

6 Experimental Evaluation

To evaluate the approach, we performed some experiments in an office environment. During the offline stage, two images are captured as reference frames. Displayed in Fig. 1 (a) is one reference frame that we captured. From these two images, features were detected using SIFT. For each feature point detected, a descriptor is calculated using the algorithm presented in Section 4. All the descriptors are then stored in a database.

Our goal was to perform 6–DOF pose tracking of a moving camera in the office environment. As shown in Fig. 1 (b), the virtual square is the one sitting on the telephone (drawn in yellow). Although the four corner points of the virtual square creates a plane, the plane does not belong to the real environment, as the surface of the telephone is a curved one. The world coordinate system is defined by the virtual square, with the center of the square as the world coordinate origin (x axis pointing to right and y axis pointing up). Once the virtual square is specified, we can locate the four corner points of it in the reference frames. These 2D coordinates are also stored in a database.

During the online tracking stage, feature points were detected in the current camera frame. Based on the descriptors built from these feature points as well as those stored in the database, the 3D camera pose can be calculated using the approach presented in Section 5. Using the calculated pose matrix $\boldsymbol{P}$, we projected a virtual cube into the real–world scene for augmentation purposes. Depending on application needs, we can also use the calculated camera pose to render other virtual objects into the real world. For the purpose of visualizing the accuracy of the registration, we have shown the augmentation as a virtual cube. As can be seen from Fig. 1 (b), the virtual cube has the virtual square as its bottom surface and is aligned with the world coordinate system.

During tracking, the augmented cube remains on the telephone, even when the environment and viewpoint changes. In Fig. 1 (b)–(c), the camera has been moved to a

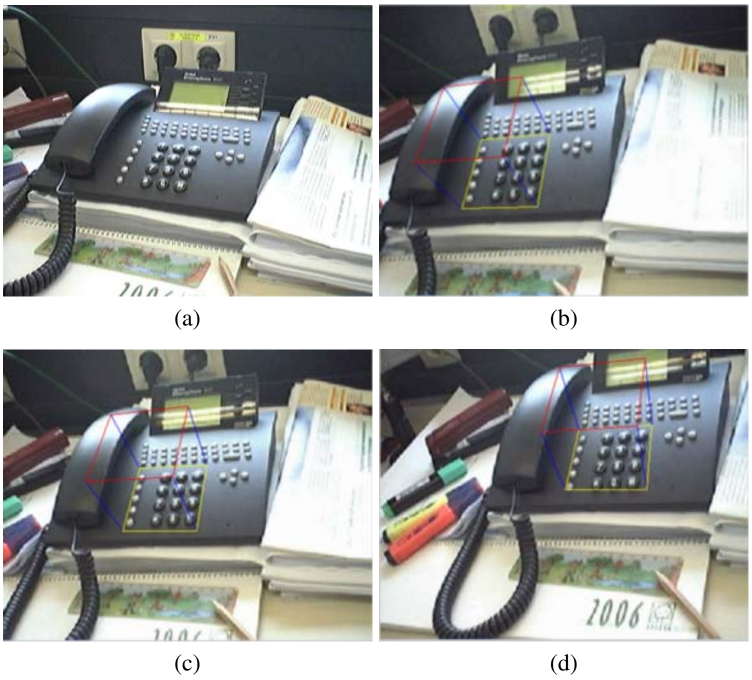


Fig. 1. Tracking and augmentation in an office environment. (a) The reference frame captured offline. (b)–(d) Online augmentation of the telephone with a virtual cube under viewpoint and environmental changes.

different pose to the one shown in Fig. 1 (a). Fig. 1 (d) illustrates that even when occlusions and the changes of viewpoint have occurred, the augmentation remains correct, which proves that pose estimation has been calculated accurately.

Further examples are shown in Fig. 2. As can be seen, despite large viewpoint changes, high degrees of occlusions, and environmental changes caused either by adding new objects into the environment or by moving the objects around, correct registration and augmentation can still be achieved due to the robust calculated camera pose.

To measure the tracking performance, we compute an "augmentation rate" which is equal to the ratio of correctly augmented frames to the total frames captured. A frame with a visually correct augmentation indicates not only the proper detection and matching of the feature points but also the correctly estimated 3D camera pose. We capture several sequences (each with about 1000 frames), containing dynamic environmental changes, occlusion as well as abrupt camera motion. In all cases, the achieved augmentation rate remains above 80%. The best rate attained was 95%.

Because we use a scale invariant feature detector, the detection of feature points has to be carried out on several scales. This results in a relatively slow detection of feature points. Fortunately, our compact feature descriptor compensates this factor. For example, matching and pose estimation take only one-third of the time used for feature detection. As a consequence, the tracking system can run at about 5 frames/s on a DELL M20 Precision laptop together with a Logitech webcam.

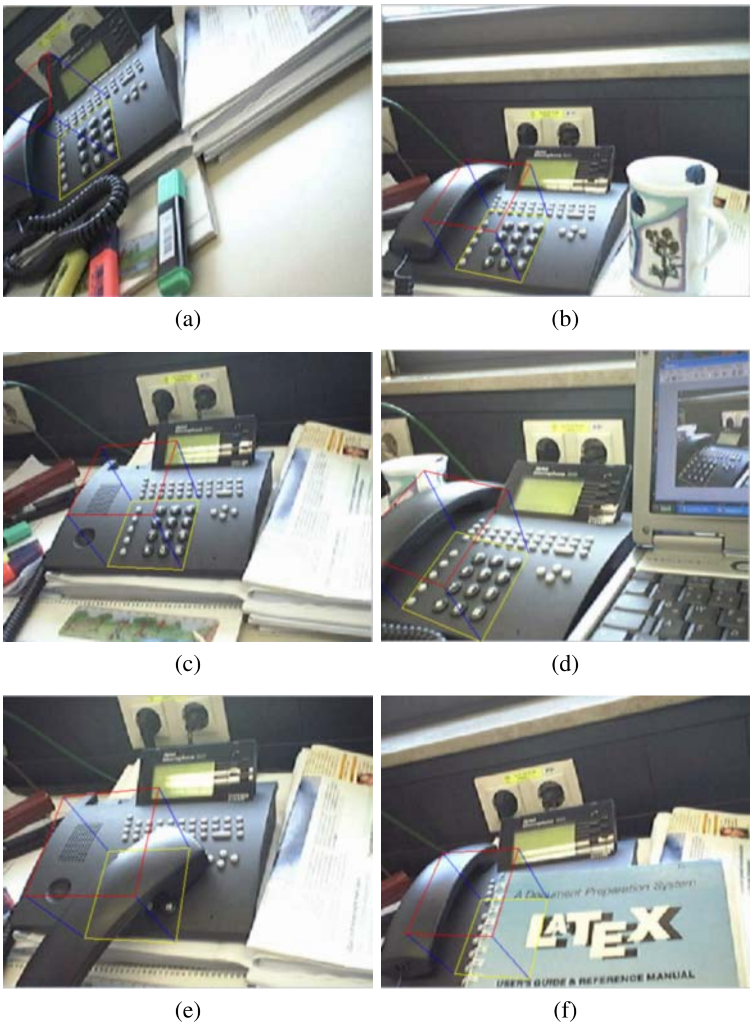

(a)
(b)
(c)
(d)
(e)
(f)
A Document Preparation System
LATEX
USER'S GUIDE & REFERENCE MANUAL

Fig. 2. Further examples of tracking and augmentation. (a) Large viewpoint change and occlusion have occurred. (b) New object comes into the environment. (c) Another scene with the position of some objects changed in the environment. (d) A different environment with new and similar objects as well as with occlusions. (e) Occlusion changes the scene substantially. (f) High–degree occlusions occur.

7 Conclusion

A new approach has been presented for 6–DOF pose tracking of a camera in an unconstrained environment. We use a localized feature descriptor for the matching of salient feature points belonging to the current camera frame with those extracted from the reference frames. A virtual square with known size is attached to the real–world scene so that a reference coordinate system is defined relative to both the real and virtual world.

With such a virtual model and the established 2D point correspondences, camera pose can be calculated using robust matching and estimation algorithms. The accurately estimated pose can be used to render any type of 3D object into the scene.

Currently we are working on the development of a new feature detector for achieving real–time performance. Detailed analysis of the tracking precision as well as comparative study with other tracking approaches (either marker–based or markerless) will be carried out shortly. In the future, we will apply this approach in several mobile and outdoor AR applications.

References

1. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge, UK (2000)
2. Szeliski, R., Kang, S.: Recovering 3D shape and motion from image streams using non–linear least squares. In: Technical Report CRL 93/3, Digital Equipment Corporation, Cambridge Research Laboratory (1993)
3. Guo, Y., Hsu, S., Samarasekera, S., Sawhney, H., Kumar, R.: Multi–view 3D analysis with applications for augmented reality and enhanced video visualization. In: IEEE Conference on Computer Vision and Pattern Recognition. (2000) 2780–2781
4. Cornelis, K., Pollefeys, M., Vergauwen, M., Van Gool, L.: Augmented reality using uncalibrated video sequences. Lecture Notes in Computer Science **2018** (2001) 144–160
5. Shi, J., Tomasi, C.: Good features to track. In: Conference on Computer Vision and Pattern Recognition, Seattle, USA (1994) 593–600
6. Hager, G., Belhumeur, P.: Efficient region tracking with parametric models of geometry and illumination. IEEE Trans. Pattern Analysis and Machine Intelligence **20** (1998) 1025–1039
7. Jurie, F., Dhome, M.: Real time robust template matching. In: The 13th British Machine Vision Conference, Cardiff, UK (2002) 123–132
8. Isard, M., Blake, A.: Condensation — conditional density propagation for visual tracking. Int. Journal of Computer Vision **29** (1998) 5–28
9. Jaynes, C., Hou, J.: Temporal registration using a kalman filter for augmented reality. In: Vision Interface, Montreal, Canada (2000)
10. Comaniciu, D., Ramesh, V., Meer, P.: Kernel–based object tracking. IEEE Trans. Pattern Analysis and Machine Intelligence **25** (2003) 564–577
11. Claus, D., Fitzgibbon, A.W.: Reliable fiducial detection in natural scenes. In: The 8th European Conference on Computer Vision. Volume 3024., Springer–Verlag (2004) 469–480
12. Kato, H., Billinghurst, M.: Marker tracking and HMD calibration for a video–based augmented reality conferencing system. In: The 2nd Int. Workshop on Augmented Reality, San Francisco, USA (1999) 85–94
13. Fiala, M.: ARTAG, a fiducial marker system using digital techniques. In: IEEE Conference on Computer Vision and Pattern Recognition, San Diego, USA (2005) 590–596
14. Harris, C., Stephens, M.: A combined corner and edge detector. In: The 4th Alvey Vision Conference, Manchester, UK. (1988) 189–192
15. Mikolajczyk, K., Schmid, C.: An affine invariant interest point detector. In: The 7th European Conference on Computer Vision, Copenhagen, Denmark (2002) 128–142
16. Lowe, D.G.: Distinctive image features from scale–invariant keypoints. Int. Journal of Computer Vision **60** (2004) 91–110
17. Ke, Y., Sukthankar, R.: PCA–SIFT: A more distinctive representation for local image descriptors. In: IEEE Conference on Computer Vision and Pattern Recognition, Washington DC, USA (2004) 506–513

18. Lepetit, V., Lagger, P., Fua, P.: Randomized trees for real–time keypoint recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, San Diego, USA (2005) 775–781
19. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Trans. Pattern Analysis and Machine Intelligence **27** (2005) 1625–1630
20. Skrypnyk, I., Lowe, D.: Scene modeling, recognition and tracking with invariant image features. In: Third IEEE and ACM International Symposium on Mixed and Augmented Reality, Arlington, VA, USA (2004) 110–119
21. Bleser, G., Pastarmov, Y., Stricker, D.: Real–time 3D camera tracking for industrial augmented reality applications. In: The 13th Int. Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, Plzen, Czech Republic (2005)
22. Lepetit, V., Vacchetti, L., Thalmann, D., Fua, P.: Fully automated and stable registration for augmented reality applications. In: Second Int. Symposium on Mixed and Augmented Reality, Tokyo, Japan (2003) 93–101
23. Comport, A., Marchand, E., Pressigout, M., Chaumette, F.: Real–time markerless tracking for augmented reality: the virtual visual servoing framework. IEEE Trans. on Visualization and Computer Graphics **12** (2006) 615–628
24. Simon, G., Fitzgibbon, A., Zisserman, A.: Markerless tracking using planar structures in the scene. In: Int. Symposium on Augmented Reality, Munich, Germany (2000) 120–128
25. Ferrari, V., Tuytelaars, T., Van Gool, L.: Markerless augmented reality with a real–time affine region tracker. In: International Symposium on Mixed and Augmented Reality, New York, USA (2001) 87–96
26. Chia, K., Cheok, A., Prince, S.: Online 6DOF augmented reality registration from natural features. In: First Int. Symposium on Mixed and Augmented Reality, Darmstadt, Germany (2002) 305–313
27. Najafi, H., Navab, N., Klinker, G.: Automated initialization for marker–less tracking: A sensor fusion approach. In: Third IEEE and ACM International Symposium on Mixed and Augmented Reality, Arlington, VA, USA (2004) 110–119
28. Koch, R., Koeser, K., Streckel, B., Evers-Senne, J.: Markerless image–based 3d tracking for real–time augmented reality applications. In: The 7th Int. Workshop on Image analysis for multimedia interactive services, Montreux, Switzerland (2005)
29. Fischler, M., Bolles, C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of ACM **24** (1981) 381–395
30. Horaud, R., Conio, B., Leboulleux, O.: An analytical solution for the perspective–4–point problem. Computer Vision, Graphics, and Image Processing **47** (1989) 33–44
31. DeMenthon, D., Davis, L.: Model–based object pose in 25 lines of code. Int. Journal of Computer Vision **15** (1995) 123–141
32. Oberkampf, D., DeMenthon, D., Davis, L.: Iterative pose estimation using coplanar feature points. Computer Vision and Image Understanding **63** (1996) 495–511
33. Schweighofer, G., Pinz, A.: Robust pose estimation from a planar target. In: Technical Report, TR–EMT–2005–01, Graz University of Technology (2005)

Lip Detection Using Confidence-Based Adaptive Thresholding

Jin Young Kim[1], Seung You Na[1], and Ronald Cole[2]

[1] Dept. of Electronics and Computer Eng., Chonnam National University
300 Yongbong-Dong, Buk-Gu, Gwangju, 500-757, South Korea
{beyondi, syna}@chonnam.ac.kr
[2] CSLR, University of Colorado at Boulder
1777 Exposition Drive, Boulder CO 80301 USA
cole@cslr.colorado.edu

Abstract. In this paper we propose a lip detector based on adaptive thresholding for hue-transformed face images. The adaptation is performed according to the confidence values of the estimated lip regions. The confidence of lip means how much similarity exists between the detected lip region and a true lip. We construct simple fuzzy rules of the confidence using true lip statistics of center position, width and height. The threshold value is adaptively changed so that the confidence of a renewed lip region is maximized. By lip detection experiments with VidTimit database we demonstrate the performance enhancement of our proposed method.

1 Introduction

Lip detection is a very important issue in audio-visual (AV) signal processing such as automatic lip reading, AV speech recognition and AV speaker recognition and authentification [1-3]. The misdetection of lips degrades the performance of lip reading and speaker identification.

Several methods of lip detection have been proposed [1, 4-8]. Those methods are classified into two groups: One is based on the active shape model (ASM) [4] and the other is to utilize color information [5-8]. ASM-based methods need more complicated calculation than color-based approaches (CBA). Thus CBAs are widely used in the areas of audio-visual signal processing. CBAs of lip detection represent the observation that lips have a much redder color than the surrounding area. Most CBAs use the color information of hue to detect lip regions. Usually pseudo hue information is used since the hue transformation needs much more calculation amount. The basic idea of lip detection, using the hue-transformed image, is to convert it into the binary image with for a given threshold. The threshold is set only for the lip region for a survival after thresholding. But this thresholding method often makes errors since faces have different color distributions for each person and different illumination conditions. Therefore, non-fixed thresholds are adopted in most CBAs. To determine the threshold, histogram or probability of hue samples is used [1, 6]. Because the size and the color of lips are dynamical variables, there are still errors even though non-fixed thresholds are adopted.

In [9] the concept of confidence was used in a person tracking problem. The estimated face regions were provided using neural networks. In other words, they calculated the confidence value of candidate images for the output of the neural network. In the previous papers the detected lips have not been tested with the concept of confidence. Thus we think that the performance of lip detection could be enhanced by applying confidence.

In this paper we propose an adaptive thresholding method for color-based lip detection. The threshold value is changed to increase the confidence value of the lip candidate region. The problem is how to measure the confidence of detected lips. In this paper, as a preliminary study, we propose a fuzzy rule based confidence measure determined from lip region statistics. Our approach is evaluated by lip detection experiments with VidTimit database [10].

2 Baseline System for Lip Detection

In this section we describe the baseline lip detector based on common image processing techniques. The baseline system of lip detection is constructed with the operations of hue transformation, the conversion of the hue image to the binary image by thresholding, RGB-to-gray transformation and thresholding, and *x* and *y* projections. Figure 1 shows the procedures of lip detection. Each procedure is as follows:

- First, an RGB colored face image is converted to the hue image. According to the reference [5], it is easier to detect lips in hue/saturation color space because hue/saturation color for the lip region is fairly uniform for a wide range of lip colors. One of our purposes is to implement a lip detector with a small calculation amount if possible. Thus we use a pseudo-hue space instead of the true hue space. The pseudo-hue is calculated by Eq (1) [5].

$$P(i,j) = \frac{R(i,j)}{R(i,j)+G(i,j)}, \tag{1}$$

  where $R(i,j)$ and $G(i,j)$ are the red and green color values of the *i-th* and *j-th* pixel, respectively and $P(i,j)$ is the pseudo-hue value of the pixel.

- Second, the histogram of the hue image is estimated against the lower area of face image, in which every possible lip is included. Then the normalized cumulative histogram is obtained as follows:

$$CH(i) = \sum_{k=1}^{i} H(k) \Big/ \sum_{k=1}^{K} H(k), \tag{2}$$

  where $H(i)$ is the histogram of the image, K is the total number of histogram bins, and $CH(i)$ is the normalized cumulative histogram. The *i-th* bin represents the number of pixels whose hue values are between V_i and $V_{i+1.}$ Then we can determine the threshold with a given percent coverage value as follows:

$$\begin{aligned} &L = \min(i) \ \textit{such that} \ CH(i) > \alpha \\ &Th = V(L) \end{aligned} \tag{3}$$

where α is the coverage threshold, and $V(L)$ is the representative value of *L-th* bin.

- Third, the hue image is converted to the binary image with the threshold obtained by Eq. (3). That is, pixels which have hue values bigger than the threshold are assigned as 1. Of course, pixels with smaller values are marked as 0.

- Fourth, the RGB image is transformed to the gray image. Then the gray image is converted to the binary image with a threshold. This procedure is added to include dark inner regions of lips into the lip area. This binary image is merged with the hue based binary image.

- Fifth, x and y projections are performed to the binary image. Let $B(i, j)$ be the matrix representation of the binary image. Then, the projections are defined by Eq. (4).

$$\begin{aligned} &Xprj(j) = \sum_i B(i, j) \\ &Yprj(i) = \sum_j B(i, j) \end{aligned} \tag{4}$$

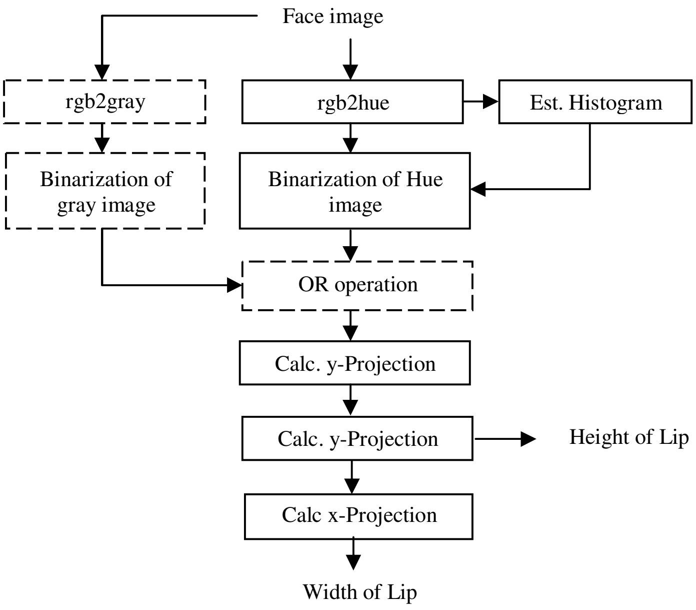


Fig. 1. Procedure of the baseline lip detection

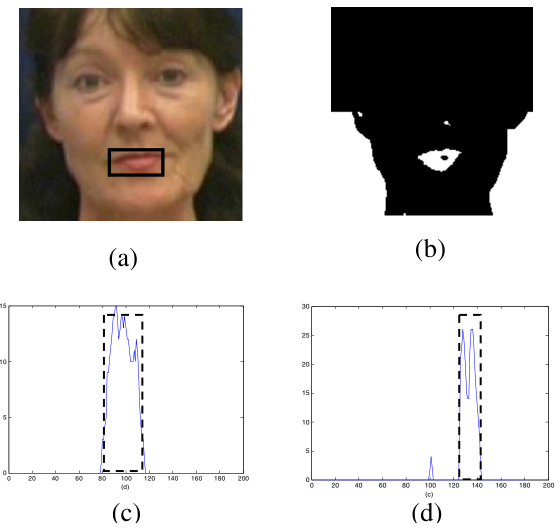


Fig. 2. Example of lip detection with the baseline system (α=0.85). (a) face image (b) binary image (c) *x*-projection (d) *y*-projection.

Lastly, the height and the width are estimated from the information *x* and *y* projections by a maximum value detection and thresholding with cut-off thresholds.

Figure 2 shows the procedure of lip detection in the case of a well-operated face image. As shown in Figure (c) and (d), we can determine the width and height of the lip by finding the main lobe of projection profiles. In many cases, however, it is not easy to detect the lip region with the fixed coverage threshold although the hue threshold is changed with the fixed coverage threshold in each face image. Figure 3 shows the case of the mis-detected lip. It is observed that the upper and lower lips in Figure 3 (b) are separated. Also, the size of the detected lower lip is smaller than that of the true lip. By the experiments with some face images the followings are observed.

(1) The lower and upper lips are separated in the binary image of the hue image. So the lower or upper lip is missing.
(2) The surround pixels of the lip region have very similar hue values to the lip's hue values. Thus the estimated lip region is larger than the true lip region.

From the above observations we conclude that it is necessary to control the coverage threshold adaptively. But it is very difficult to change the coverage threshold adaptively, because in a real application we don't know whether the detected lip is correct or not. Thus, to adaptively change the threshold, we have to have a measure with which the confidence of the detected lip can be estimated. If we can calculate a confidence of the detected lip, we can develop a confidence maximization approach for lip detection. In section 3, we will discuss our proposed confidence measure and its application to lip detection.

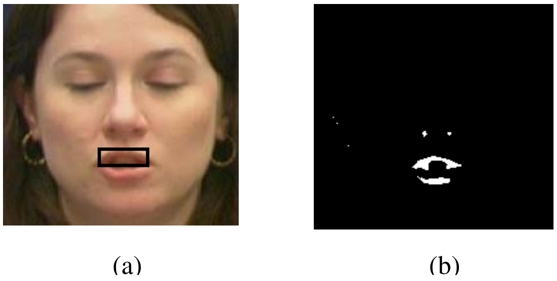


Fig. 3. Example of mis-detected lip. (a) face image (b) binary inage

3 Adaptive Thresholding Based on Confidence

In section 2 we discussed the problems with the previous thresholding method. In this section we describe our proposition of an adaptive thresholding based on confidence. First, let's discuss the main ideas of our approach. After some experiments we observed the followings in the case of mis-thresholding.

(1) The center point (C_x,C_y) of the detected lip is far from the center of the true lip.
(2) The width, height or size of the detected lip region is larger or smaller than the true lip region.

Our approach is to use the above observations for developing a confidence of a detected lip. With what method can we measure how the detected lip is different from the true one? Our idea is to use the statistics of the true lips in face images. Even though the hue distributions of lips are different from each other, the geometrical positions and sizes of lips are similar for people. Thus it makes sense to use the lip geometrical statistics as a candidate of confidences.

To gather the information of true lips, first, we performed the face detection based on boosting algorithm [11]. By the algorithm in [11] we extracted the squares of face regions from VidTimit database images. And then we marked the true lip region manually for about 1,000 face images. Table 1 shows the statistics of lip regions obtained from the lip images. The parameters are normalized so that the width and the height of detected face image have the unit length in Table 1. Note that the mean of Center$_x$ is not 0.5. The reason, we think, is that the detected faces are a little biased. That is, this error is generated by the face detector. The statistics of Table 1 reflect the error made by the face detector.

Using the data in Table 1, we can calculate the probability of the detected lip. And then we can change the coverage threshold so that the detected lip region's probability can be maximized. The problem can be represented as follows:

$$Th_C^* = \max_{Th_C} Prob(R_{Lip}(Th_C)) \tag{5}$$

where Th_C is the coverage threshold and R_{Lip} is the detected lip region.

Table 1. Geometrical statistics of true lips in face images

	Center$_x$	Center$_y$	Width	Height
Mean	0.530	0.727	0.289	0.150
Standard deviation	0.020	0.030	0.035	0.036

But there is a problem when the formulation (5) is used as a maximization criterion. Figure 4 shows the graphs of lip statistics. Because the standard deviation of lip width is 0.035, the lip region of which the width is in the range of [0.289-0.035*3, 0.289+0.035*3] is possibly a lip. According to Figure 4 (b), the maximum value is 11.38 and the probability at the value of (μ+3σ) is 0.1266. If we use lip statistics as a maximization criteria, lip detection almost results in lip regions with the average lip statistics. Thus we conclude that the probability density functions of lip statistics cannot be used as a confidence function. Therefore we transform the pdfs into fuzzy membership functions using a simple clipping method, as shown in (6).

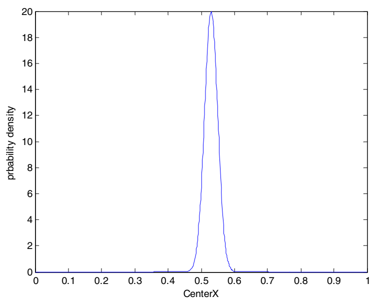
$$\mathrm{M}(x) = \begin{cases} 1, & \text{if } pdf(x) > T \\ \dfrac{pdf(x)}{pdf(T)}, & \text{otherwise} \end{cases}, \tag{6}$$

where $\mathrm{M}(x)$ is a membership function, $pdf(x)$ is a probability density function of Center$_x$, Center$_y$, width or height, and T is a threshold for transferring *pdf* into membership function.

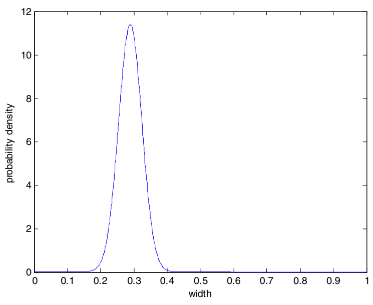
Figure 5 shows the example of width's membership function. As shown in the figure, the membership function has the value of unity in the range where the detected lip can be considered a reasonable result.

Now, we can redefine our problem by the membership functions as follows:

$$Th_C^* = \max_{Th_C} M(R_{Lip}(Th_C)) \tag{7}$$



(a) Center$_x$



(b) Width

Fig. 4. Probability density functions of lip statistics

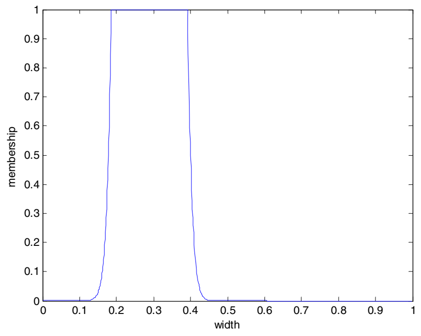


Fig. 5. Membership function of width (T=0.15)

The problem is to find the coverage threshold and the lip candidate region so that the membership function is maximized. On the other hand, the problem to be considered is how we can decide the direction of the threshold change. We assume that the membership value of the candidate is low. Then we should increase or decrease the threshold. What is the criterion for that? To control the threshold, we adopt the size information of the candidate lip.

$$Th_C = \begin{cases} Th_C - \delta, & \text{if the size of the detected lip is less than } th_{size\_low} \\ Th_C + \delta, & \text{if the size of the detected lip is less than } th_{size\_high} \\ Th_C, & \text{otherwise} \end{cases}, \tag{8}$$

where δ is a step size and $th_{size\_low\{high\}}$ is the threshold for breaking the threshold adaptation. Figure 6 shows our proposed algorithm in this paper. In the figure is th_{M1} and th_{M2} are the stopping thresholds for escaping from the iteration. As shown in the figure the coverage threshold is adaptively changed depending on lip confidence represented by membership functions.

```
$Th_C$=THC;
delta=DELTA;

for i=1:Maxiter
    1) Estimate lip region $R_{lip;}$ // (section 2)
    2) Calculate the membership functions of $R_{lip}$; // Eq. (6)
    3) If $M_{center}>th_{M1}$ and $M_{size}>th_{M2}$, break;
       else, change $Th_C$; // (Eq. 8)
end
```

Fig. 6. The proposed lip detection algorithm

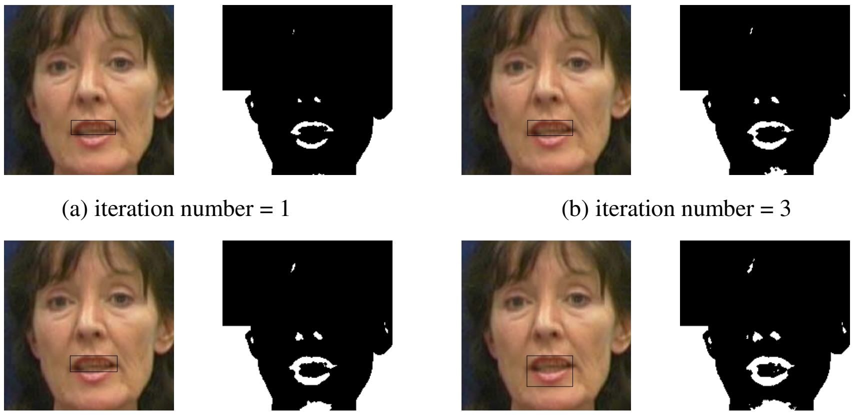


Fig. 7. Iterative lip detection with an adaptive threshold

4 Experimental Results and Discussion

We tested our algorithm with several face images of VidTimit database. The VidTimit database consists of video and corresponding audio recordings of 43 people (19 female and 24 male), reciting short sentences selected from the NTIMIT corpus.

The data were recorded in three sessions, with a mean delay of seven days between Sessions 1 and 2, and of six days between Sessions 2 and 3. In this paper we selected the data of 20 persons (10 female and 10 male) from the database. We tested the proposed method to the images, which are about five percent of these face image frames, that is, that is, approximately a thousand face images were tested.

Figure 7 shows the results of performing our algorithm. As shown in Figure 7, the detection of lips gets correct after seven iterations. In this experiment Th_C and delta are set to 0.83 and 0.003, respectively. Maxiter number in Figure 6 is chosen as 10. Observing the binary images in each step, we find that the number of white pixels increases as the coverage threshold decreases. Figure 8 shows the changes of membership functions of lip center and size with the iteration. From Figure 8 we can observe that in the first two iterations no enhancement is achieved. But the membership values of lip center and size converge to 1 from the third iteration. Comparing Figure 7 (d) with Figure 7 (a), the widths of the upper and lower lips get thick. So, according to the examples shown in Figure 7, the widths of the upper and lower lips are more accurately estimated using our algorithm, if this information is targeted. Because our purpose in this paper is to detect lip regions, we do not perform any specific lip parameter estimation.

Table 2 shows the performance of the proposed lip detector. According to Table 2, the performance enhancement of a 5.5% detection rate has been achieved with an adaptive thresholding based on confidence.

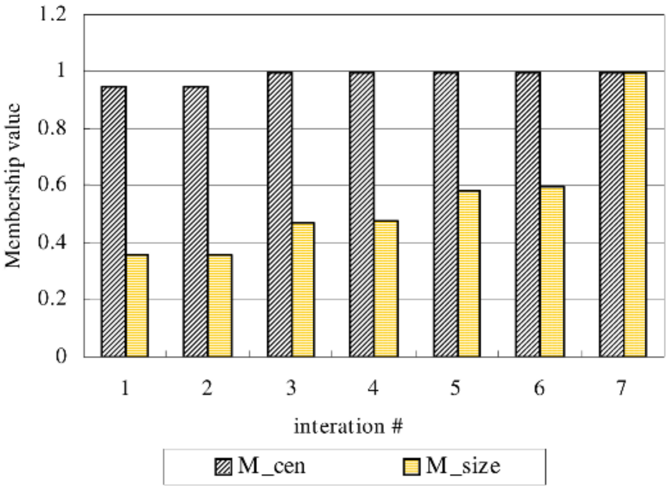


Fig. 8. Change of membership values with iteration number (M_cen : lip center membership, M_size : lip size membership)

Table 2. Detection ratio of lip without and with an adaptive thresholding

	Without adaptive thresholding	With adaptive thresholding
Detection rate	91.3%	96.8%

The results shown in Table 2 are preliminary. We will perform the experiments with the whole VidTimit face images. Anyhow we think that our proposed approach was successfully evaluated.

Even though we achieved the improvement of lip detection, we found that some problems need to be solved in the future.

(1) The dark regions estimated from the gray image include not only the regions surrounded by the upper and lower lips but also beard or mustache areas. In that case, the gray-level based inner region detection should be excluded.
(2) The step size of delta decides the convergence speed of our algorithm. So we need more experiments on the parameter setting.

We will cope with the problems revealed by these experiments in near future.

5 Conclusion

In this paper we proposed an adaptive thresholding method for lip detection. To construct our algorithm, we suggested a confidence function measuring how accurately the lip detection is performed. Also, we introduced the concept of maximizing confidence measure in the lip detection area. The experimental results showed that the proposed algorithm is very promising in lip detection, allowing not only accurate lip detection but also a small calculation amount. We only use the operations of thresholding and projections.

However, there needs to be further research in the future. First, our algorithm needs to be verified with many other face images. Second, more reasonable confidence functions should be developed, especially based on lip's symmetricity and roundness. Lastly, lip confidence could be adopted as a reference parameter in AV speech and speaker recognition. That is, detected lips with low confidence can be discarded from the AV recognition procedure. In the future study, we will evaluate our approach in the AV recognition area.

Acknowledgement

This paper is supported by the 2005 sabbatical grant of Chonnam National University.

References

1. Peter C, Zelensky M.: Using of Lip-Reading for Speech Recognition in Noisy Environments, Proc of the 13th Czech-German Workshop on Speech Processing (2004)
2. Gowdy, J.N.; Subramanya, A.; Bartels, C., Bilmes, J.; DBN Based Multi-stream Models For Audio-visual Speech Recognition. Proc of ICASSP'04 (2004) Vol. 1, 17-21
3. Tieyan Fu; Xiao Xing Liu; Lu Hong Liang; Xiaobo Pi; Nefian, A.V.: Audio-visual speaker identification using coupled hidden Markov models. Proc of ICIP03 (2003), Vol3, 14-17
4. Caplier A.: Lip Detection and Tracking. Proc. of 11th International Conference on Image Analysis and Processing (2001), 8-13
5. Chetty G, Wagner M.: Automated Lip Feature Extraction. Proc. Image and Vision Computing (2004) 17-22
6. Zhang J.M, Tao H, Wang L.M., Zhan Y.Z.: A Real-time Approach to the Lip-motion Extraction in Video Sequence, Proc of 2004 IEE International Conference on Systems, Man and Cybernetics (2004), 6423-6428
7. Luthon F., Leivin M.: Lip Motion Automatic Detection, Proc of Scandinavian Conference on Image Analysis (1997), Vol 1, 253-260.
8. Jacek C., Wojde l., Rothkrantz J.M.: Using Aerial and Geometric Features in Automatic Lip-reading, Proc of Eurospeech02 (2002), Vol.4, 2463-2466
9. Kim J.Y., Song M.G, Na S.Y., Baek S.J, Choi S.H, Lee J.H.: Skin-Color Based Human Tracking Using a Probabilistic Noise Model Combined with Neural Network, LNCS Vol. 3972, Springer-Verlag Berlin Heidelberg New York (2006), 419-428.
10. Sanderson C.: VidTimit database documentation. http://users.rsise.anu.edu.au/~conrad/vidtimit/
11. Fasel I, Fortenberry B., Movellan J.: A Generative Framework for Real Time Object Detection and Classification. Computer Vision and Image Understanding (2005), Vol. 98, 182-210

Optic Flow Integration at Multiple Spatial Frequencies - Neural Mechanism and Algorithm

Cornelia Beck, Pierre Bayerl, and Heiko Neumann

Dept. of Neural Information Processing, University of Ulm, Germany
{cornelia.beck, pierre.bayerl, heiko.neumann}@uni-ulm.de

Abstract. In this work we present an iterative multi-scale algorithm for motion estimation that follows mechanisms of motion processing in the human brain. Keeping the properties of a previously presented neural model of cortical motion integration we created a computationally fast algorithmic implementation of the model. The novel contribution is the extension of the algorithm to operate on multiple scales without the disadvantages of typical coarse-to-fine approaches. Compared to the implementation with one scale our multi-scale approach generates faster dense flow fields and reduces wrong motion estimations. In contrast to other approaches, motion estimation on the fine scale is biased by the coarser scales without being corrupted if erroneous motion cues are generated on coarser scales, e.g., when small objects are overlooked. This multi-scale approach is also consistent with biological observations: The function of fast feedforward projections to higher cortical areas with large receptive fields and feedback connections to earlier areas as suggested by our approach might contribute to human motion estimation.

1 Introduction

The detection of motion in our environment is part of our everyday life. Humans are capable to estimate motion very precisely and very fast. While ongoing research tries to resolve the detailed processing mechanism in the human brain, it is agreed on that areas V1, MT and MST play important roles in the motion processing pathway [1]. In contrast to the simple task of motion detection for a human, implementing motion detection in technical applications remains a difficult problem (see [2] for an overview of existing technical approaches). For instance, a moving robot provided with cameras should be capable to detect moving objects in its environment in real-time to avoid collision. Therefore, the motion estimation has to be fast, of high quality, and the motion estimates should be available for every position of the surrounding.

We developed a model for motion estimation that is based on the mechanisms observed in human motion processing (see Sect. 2) [3]. This neural model simulates areas V1 and MT, including feedforward as well as feedback connections [4]. To shorten the computing time, we reimplemented the model in an algorithmic version [5] and improved its results by adding multiple processing scales as described in Sect. 3. Furthermore, we will explain the biological motivation of this new approach.

2 Neural Model

Approaches in computer vision for optic flow estimation use, e.g., regularization or Bayesian models to achieve globally consistent flow fields [6,7]. Another possibility to approach this problem is to build a model corresponding to the neural processing in our visual system. We previously presented such a model for optic flow detection based on the first stages of motion processing in the brain [3]. Therein, areas V1 and MT of the dorsal pathway are simulated. In model area V1 a first detection of optic flow is realized, model area MT estimates the optic flow of larger regions and is thus, e.g., capable to solve the aperture problem [8]. The principle processing components of this neural model are feedforward, feedback and lateral connections.

Both modules V1 and MT comprise at each spatial location a certain number of neurons tuned to different velocities. For efficient computation, we need to discretize and limit the velocity space. Each neuron has a certain activity rate describing the likelihood of its represented velocity. The input net_{IN} for module V1 of the model represents the similarity of the image structure of two images at different time steps that is calculated using modified Reichardt detectors [9]. It is modulated with the feedback net_{FB} from module MT (1). This multiplicative feedback only enhances activated neurons, but will not create new activities. In the process of feedforward integration, signal $v^{(1)}$ is integrated with Gaussian isotropic filters in both the velocity and the spatial domain (2), "*" denotes the convolution operation. Finally, lateral shunting is effected at each location to strengthen the activity of unambiguous motion signals (3). The equations are identical for module MT, but the integration process (2) uses a larger spatial neighborhood and there is no feedback to MT.

$$\delta_t v^{(1)} = -v^{(1)} + net_{\text{IN}} \cdot \left(1 + C \cdot net_{\text{FB}}\right) \ . \tag{1}$$

$$\delta_t v^{(2)} = \left(v^{(2)}\right)^2 + v^{(1)} * G_{(\text{space})} * G_{(\text{velocity})} \ . \tag{2}$$

$$\delta_t v^{(3)} = -0.01 \cdot v^{(3)} + v^{(2)} - \left(1/\left(2n\right) + v^{(3)}\right) \cdot \sum_{\Delta x} v^{(2)} \ . \tag{3}$$

3 Algorithmic Multi-scale Model

A technical problem of the neural algorithm is the fact that the high number of neurons leads to large memory costs and to long simulation times. Thus, starting from the neural model for optic flow estimation, we have previously developed an efficient algorithmic version that behaves similar to the neural model [5].

A limitation of the neural as well as of the algorithmic model is that the optic flow is only evaluated on a single scale, i.e., the similarities in the input images are calculated on a single spatial resolution of the input image. This leads to problems when spatially low-frequency areas are moving in an image sequence. On a fine scale, this "coarse" motion may not be detected, as there exist many ambiguities when calculating the similarity measure for V1 between pairs of frames (see example in Fig. 2). To solve this problem we extend the algorithmic

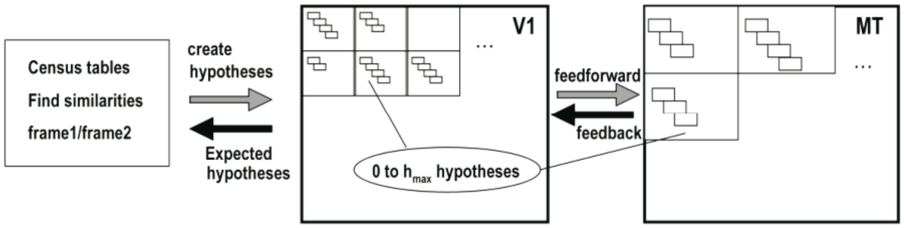


Fig. 1. Iterative model of optic flow estimation: First, the similarity of two input frames is calculated using the Census values. The hypotheses of V1 influence the creation of the initial optic flow hypotheses, if the number of Census values is bigger than $h_{\max}$. Locations in V1 are spatially integrated and subsampled for the optic flow estimation of MT. The *feedback* of MT modulates the likelihood of hypotheses in V1.

model with coarser scales of motion estimation. Such a "multi-scale processing" scheme was proposed, e.g., by Simoncelli [10]. In general, these algorithms are realized with "image pyramids" where motion estimation of coarser scales influences the estimation of finer scales as "initial guess" [10,11]. The processing of the input image in resolutions of different spatial frequencies provides more information for the motion estimation.

Considering the biological basis of our model, coarser processing scales can be integrated in a plausible way. We believe that fast motion processing on a coarser scale can modulate the feedforward projection within the fine scale of motion estimation from V1 to MT [1]. This can be accomplished via feedback of the processing of a coarser representation of the input image from area V1 to MST. Our approach is in line with the "facilitation" of visual object recognition in human brain via expectations in the prefrontal cortex which act as "initial guess" as proposed by Bar [12]. In the following subsection we will explain the algorithmic version of the neural model with one processing scale. Thereafter, the extension of this model from one scale to two and more scales is presented, that combines fast optic flow estimation with improved qualitative performance due to the integration of multiple scales.

3.1 Algorithmic Single-Scale Model

The algorithmic model consists of two different modules V1 and MT like the neural model (see Fig. 1). For the extraction of motion correspondences between two frames of an image sequence the algorithm uses a similarity measure of the class of rank-order approaches: The "Census transform" [13] provides an implicit local description of the world. Accordingly, possible motion correspondences between two frames of an image sequence can be extracted at locations with the same Census values in both frames. Initially, we extract motion correspondences (hypotheses) in V1 for identical Census values which show less than $h_{\max}$ possible correspondences in the second frame. Each hypothesis includes a likelihood initialized to 1.0 which indicates the likelihood of a particular velocity

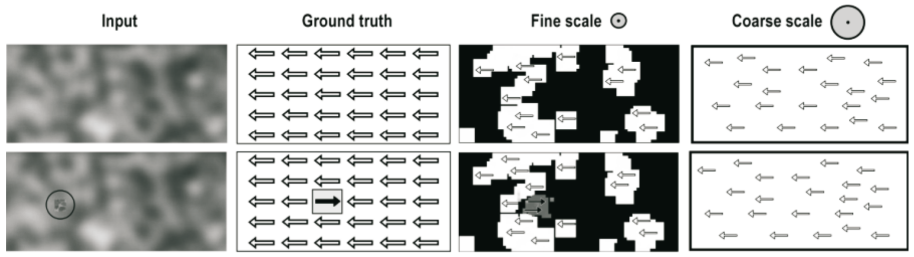


Fig. 2. Optic flow detection with the algorithmic model with one scale. First row: Results for an input image containing a spatially low frequency structure. Second row: Results for an input image with a small moving object in front of the moving background. The object is indicated by the black circle (not part of the input image). The second column represents the ground truth of the input. In the third and the fourth column the resulting flow fields of MT are shown after the first iteration using a fine and alternatively a coarse scale. Black positions indicate positions without motion estimation, white positions represent movement to the left, gray positions to the right. One iteration of the fine scale algorithm takes about 0.6 seconds, using the coarse scale, one iteration takes less than 0.1 seconds (Pentium IV, 3GHz).

at a certain position. The restriction on $h_{\max}$ identical Census values results in at most $h_{\max}$ correspondences selected at each pixel (we use $h_{\max} = 4$). This means, in comparison to the neural model, that we only simulate the $h_{\max}$ neurons representing the velocities with the highest likelihood at each position. The extracted motion hypotheses include a lot of wrong hypotheses in addition to the correct ones caused, e.g., by the motion aperture problem. To improve the estimations, the hypotheses of the first module are spatially integrated to generate the hypotheses for the second module MT, which represents motion at a coarser spatial resolution. Hypotheses that are supported by adjacent positions have an advantage during the subsampling process in comparison to spatially isolated motion hypotheses. Again, only the $h_{\max}$ hypotheses with the strongest likelihood are kept at each position in the second module. These hypotheses are iteratively used as feedback for motion estimation in the first module. The recurrent signal modulates the likelihood of predicted hypotheses in the first module as in the neural model. In addition, the feedback influences the input to V1: Hypotheses are also created at positions with ambiguous motion hypotheses (Census value appears more than $h_{\max}$ times in the second image) if the velocity at this position corresponds to one of the velocities of the feedback (i.e., the velocity is expected). This procedure is necessary in the algorithmic model to compensate that only $h_{\max}$ hypotheses are represented at a position in contrast to the neural model where each possible velocity is computed at a position.

3.2 Integration of Multiple Scales

In the first row of Fig. 2 the results of the single-scale algorithm are presented for an input image (320x144 pixel) consisting of a spatially low-frequency texture like clouds that are moving to the left. Only few hypotheses are generated in V1 and MT when using the algorithm on the fine scale. In comparison to this, using the same algorithm but a coarse version of the input images, all the positions of MT show (the correct) motion hypotheses. This is due to the fact that the movement in the coarse scale is less ambiguous.

However, single-scale models operating only on a coarse scale have another disadvantage: Small moving objects with minor luminance contrast are overlooked by the model, as they are effaced during the subsampling and the motion integration process. An example is given in the second row of Fig. 2, where a small rectangle (17x17 pixel) is moving to the right in front of a background moving to the left. Whereas the model using the coarse scale completely overlooks the objects after the first iteration, the object is detected in the fine scale.

To combine the advantages of the fine and the coarse scale we need to integrate the feedback of at least one coarser scale (e.g., V1-MST) to our single-scale model. In doing so, the estimations of the fine scale need to be protected. For this reason, the coarser scales in this algorithm do only contribute to the estimation of motion in the next finer scale where the ambiguity is high. The calculation of motion estimation in a model with two scales will be calculated by the following way: In a first step, motion hypotheses of the coarse scale are created. Thereafter, the motion correspondences in V1 of the fine scale are calculated. Just if the motion at a position is ambiguous (i.e., more than $h_{\text{max}} = 4$ hypotheses), the feedback of the coarse scale is used for the selection of possible motion hypotheses. Thereby it is combined with the feedback of MT of the fine scale (i.e., both modules contribute to the creation of new hypotheses at ambiguous positions). Adding more scales can be realized in an analog way. A technical detail to keep fast processing in the algorithm that has to be considered is that the resolution of the feedback from the coarse scale has to be in the same resolution as the motion hypotheses of the module receiving the feedback (fine scale). This can easily be realized if the motion hypotheses of the coarse scale are created of frames with greater temporal distance Δt (subsampling rate (fine scale to coarse scale) corresponds to Δt). In the neural model this would not be necessary.

4 Results

The following results are obtained with a multi-scale model of the presented algorithm containing two scales of motion detection. In the coarse scale the input images processed are four times smaller than the original ones. For this subsampling process, the image is blurred with a Gaussian filter ($sigma = 4$). The integration of motion hypotheses of the second module of the coarse scale (module MST) reduces its size to a fourth of its subsampled input image from V1. The images shown here always represent the hypotheses of MT (the second module of the fine scale), as these are the final results of the motion estimation. First, we tested the developed multi-scale model with spatially low-frequency image sequences where the motion is not detected in the fine scale (see Fig. 2/first row). The detected motion hypotheses of the multi-scale model are presented

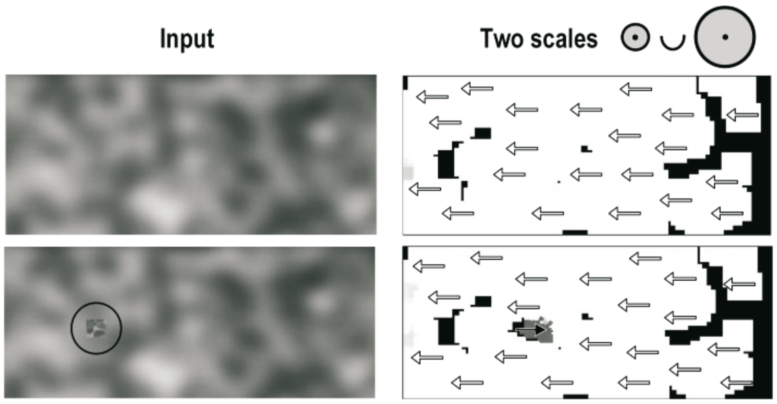


Fig. 3. Motion hypotheses of MT of the two scale model. The input images (first column) are the same as in Fig. 2. In the second column the motion hypotheses of module MT are shown. Black positions are positions where no movement was detected, white positions indicate movement to the left, gray positions represent movement to the right. One iteration of the multi-scale algorithm takes about 0.7 seconds.

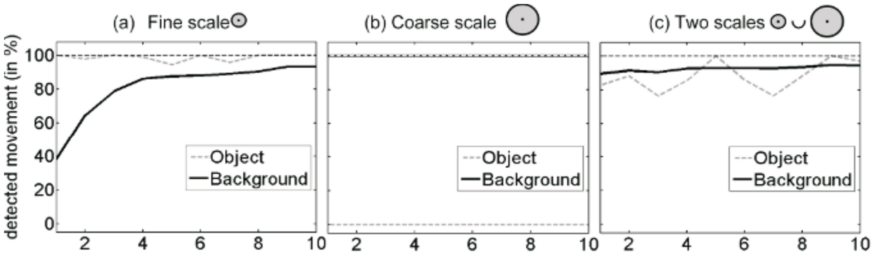


Fig. 4. Detection of background and object for input image sequence with moving object. The percent of the positions of the background and the object with the correct detected direction are shown. The dashed line at 100 percent represents the reference. **(a)** The fine scale model detects the movement of the complete object after the first iteration, but it needs 4 iterations to cover the main parts of the background. **(b)** shows the results for the coarse scale model that detects the background movement immediately, but completely misses the object, independently of the number of iterations. **(c)** In the two scale model about 85 percent of the background movement is detected after one iteration as well as the main parts of the object.

in Fig. 3/first row. In contrast to the results of the fine scale model nearly all positions represent a motion hypothesis. The direction of the hypotheses is also correct. The result is close to the optimal result of the coarse scale model. Second, the image sequence with a small moving object not detected within the coarse scale model was used as input to the multi-scale model (see Fig. 2/ second row). Whereas the coarse scale model ignores the movement of the small

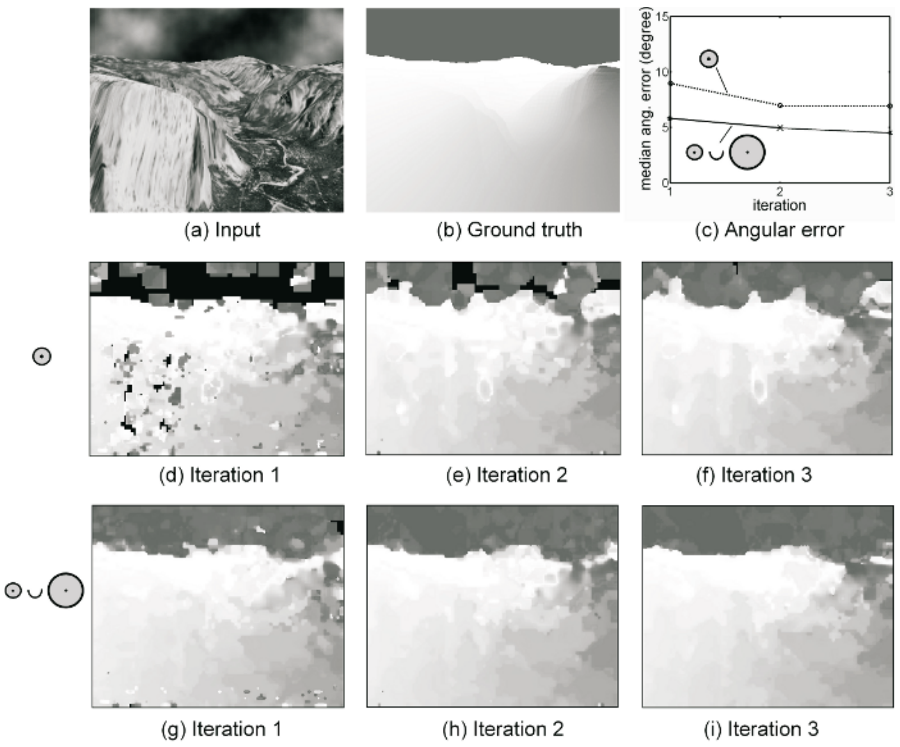


Fig. 5. **(a)** One input image of the Yosemite Sequence ($t = 3$), **(b)** ground truth for the optic flow of frame 3 and 4. For the motion in the sky we assume horizontal motion to the right as proposed in [2]. **(c)** shows the median angular error in degree of module MT. The motion estimations in MT for the Yosemite sequence using the fine scale algorithm are shown in the second row. **(d)** After the first iteration a lot of positions in the sky do not contain a motion hypothesis (black positions). Furthermore, there are some wrong motion hypotheses in the lower left. **(e)** After the second iteration the positions representing motion hypotheses in the sky is higher, the flow field contains less errors. **(f)** After three iterations more than 98 percent of the positions represent motion hypotheses, the flow field is similar to the ground truth. In the third row the results for the multi-scale algorithm are presented. **(g)** After the first iteration almost every position holds a motion hypothesis, even in the coarse structure of the sky the movement to the right is correctly indicated. The flow field contains only few errors. **(h)(i)** the positions disturbing the smoothness of the flow field are corrected in the second and third iteration.

object opposite to the background, the multi-scale model clearly indicates a rectangle in the center of the image moving to the right as depicted in Fig. 3/second row. The proportion of detected movement at the positions of the object and the background in the input image is shown in Fig. 4. Only the multi-scale model is close to 100 percent detection for both the background and the object after only one iteration.

We further compared our multi-scale approach to the fine scale model using the Yosemite Sequence (316x256 pixel, version with clouds) as input images [2]. An exemplary image of this sequence is presented in Fig. 5(a). For this artificial sequence the ground truth of the optic flow is provided which enables us to evaluate the quality of extracted motion hypotheses (see Fig. 5(b)). Gray positions represent movement to the right, white positions movement to the left, at black positions no motion hypothesis is created. The results of module MT of the fine scale model are presented in Fig. 5(d)-(f) for three iterations of the algorithm. After the first iteration the positions in MT representing at least one motion hypothesis add up to 85 percent. Similar to the texture in input image of Fig. 2 the spatially low-frequency texture of the sky causes problems to the fine scale model. Thus, mainly positions in the sky do not have motion hypotheses (black positions). At the same time, in comparison to the smooth original flow field (see Fig. 5(b)), there are some wrong hypotheses especially in the lower left area of module MT caused by the aperture problem when using only a small scale.

The motion hypotheses of MT for three iterations of the multi-scale algorithm are shown in Fig. 5(g)-(i). Just after one iteration, 98 percent of the MT positions represent motion hypotheses. The flow field contains only few positions which represent motion hypotheses that differ from the smooth flow field of the image. The aperture problem is solved due to the coarse scale added. Thus, only one iteration of the multi-scale algorithm provides a flow field comparable to the ground truth for nearly every position. A comparison of the quality of the motion hypotheses of the two algorithms is presented in Fig. 5(c). The multi-scale model achieves better results in the median angular error of the motion hypotheses than the fine scale model in MT. The better results of the multi-scale model after the first iteration is even more significant, if we take into account that 98 percent of all positions in MT of the multi-scale model cause a lower error than the 85 percent of positions with hypotheses of the fine scale model.

5 Discussion and Conclusion

We presented a multi-scale algorithm for optic flow estimation based on a neural model. In contrast to other multi-scale approaches [10], this algorithm does not propagate the error of coarser scales to the fine scale. This is ensured by the way coarse scales influences the motion estimation in the fine scale. Only if the estimation in the fine scale is highly ambiguous and if the motion estimation of the coarse scale is compatible with the motion correspondences in the fine scale, then the additional information of the large scale will be used. This avoids that small objects are overlooked on the fine scale [10]. Moreover, the extraction of the motion hypotheses is implemented in a way that the search space for corresponding positions in each scale comprises the entire image. This is not realized in many other multi-scale approaches [2].

In the examples presented for the multi-scale model we restricted the model to two scales. This is due to the fact that for the employed input sequences a combination of one coarse and one fine scale was sufficient to get estimations for all positions after one iteration. Thus, adding coarser scales would not further improve the estimations. Nevertheless, the algorithm can be extended to more scales in a straightforward way. Concerning the computing time of the algorithm, more scales do not increase the time for one iteration considerably. This follows from the faster processing of coarser scales where estimations for less positions have to be calculated. Furthermore, the time for the calculation could be further reduced by limiting the positions of the images to be processed. This could be done by a preclassification (e.g., corners [13]) or a limitation to positions with a certain minimum contrast.

The biological motivation of the multi-scale model is based on the observations that V1, MT and MST are main components of the motion processing pathway that includes feedforward and feedback connections [4]. While in V1 motion is detected only in a very small neighborhood, its projections to MT lead to an integration of the detected motion within a larger region [14]. Motion estimation in an even coarser spatial resolution is accomplished in MST, its neurons respond to planar, circular, and radial motion as well as to complex patterns of motion [15]. Low latencies of the first responses in V1, MT *and* MST [16] indicate a possible computation of a quick initial guess in higher areas, such as MST, which may in turn influence information processing in earlier areas such as V1 or MT via feedback connections. Because area MST receives its primary afferent inputs from area MT, such a computation may probably be realized in a feedforward manner via MT, but also direct connections from V1 to MST have been observed [1]. The prediction through a fast and spatially coarse "initial guess" is compatible to theories predicting that the context (here a large spatial context of motion information) may influence initial feature extraction [17,12].

In conclusion, we presented a biologically motivated algorithm for optic flow integration on multiple processing scales that generates fast and reliable motion estimations.

Acknowledgements

This research has been supported in part by a grant from the European Union (EU FP6 IST Cognitive Systems Integrated project: Neural Decision-Making in Motion; project number 027198).

References

1. L.G. Ungerleider, J.V. Haxby, 'What' and 'where' in the human brain, Current Opinion in Neurobiology, 4, pp. 157-165, 1994
2. S.S. Beauchemin, J.L. Barron, The Computation of Optical Flow, ACM Computing Surveys, Vol. 27, no. 3, pp. 433-467, 1995
3. P. Bayerl, H. Neumann, Disambiguating Visual Motion through Contextual Feedback Modulation, Neural Computation, 16(10), pp. 2041-2066, 2004
4. J.M. Hupé, A.C. James, P. Girard, S.G. Lomber, B.R. Payne, J. Bullier, Feedback Connections Act on the Early Part of the Responses in Monkey Visual Cortex, J. Neurophys., 85, pp. 134-145, 2001

5. P. Bayerl, H. Neumann, Towards real-time: A neuromorphic algorithm for recurrent motion segmentation, Ninth International Conference on Cognitive and Neural Systems (ICCNS '05), Boston, USA, 2005
6. B.K.P. Horn, B.G. Schunk, Determining optical flow, Artificial Intelligence, 17, pp. 185-203, 1981
7. Y. Weiss, D.J. Fleet, Velocity likelihoods in biological and machine vision, Probabilistic models of the brain: Perception and neural function, pp. 81-100, Cambridge, MA:MIT Press, 2001
8. C.C. Pack, R.T. Born, Temporal dynamics of a neural solution to the aperture problem in cortical area MT, Nature, 409, pp. 1040-1042, 2001
9. E. Adelson, J. Bergen, Spatiotemporal energy models for the perception of motion, Optical Society of America A, 2/2, pp. 284-299, 1985
10. E. Simoncelli, Course-to-fine Estimation of Visual Motion, IEEE Eighth Workshop on Image and Multidimensional Signal Processing, Cannes France, Sept. 1993
11. P.J. Burt, E.H. Adelson, The Laplacian Pyramid as a Compact Image Code, IEEE Transactions On Communications, Vol. 31, No. 4, 1983
12. M. Bar, A Cortical Mechanism for Triggering Top-Down Facilitation in Visual Object Recognition, Journal of Cognitive Neuroscience, 15:4, pp. 600-609, 2003
13. F. Stein, Efficient Computation of Optical Flow Using the Census Transform, DAGM-Symposium 2004, pp. 79-86, 2004
14. T.D. Albright, Direction and orientation selectivity of neurons in visual area MT of the macaque, J. Neurophys., 52, pp. 1106-1130, 1984
15. C.J. Duffy, R.H. Wurtz, Sensitivy of MST Neurons to Optic Flow Stimuli. I. A Continuum of Response Selectivity to Large-Field Stimuli, J. Neurophys., 65, pp. 1329-1345, 1991
16. V.A.F. Lamme, P.R. Roelfsema, The distinct modes of vision offered by feedforward and recurrent processing, Trends Neurosci., 23, pp. 571-579, 2000
17. A. Torralba, A. Oliva, Statistics of natural image categories, Network: Comput. Neural Syst., 14, pp. 391-412, 2003

A Critical Appraisal of the Box Counting Method to Assess the Fractal Dimension of Tree Crowns

D. Da Silva[1,5], F. Boudon[2,5], C. Godin[2,3,5], O. Puech[4,5], C. Smith[4,5], and H. Sinoquet[6]

[1]Université de Montpellier II [2]CIRAD [3]INRIA [4]INRA
[5]Virtual Plants Team, UMR AMAP TA/40E 34398 Montpellier Cedex 5, France
[6]INRA-UBP, UMR PIAF,
Domaine de Crouelle, 234 avenue du Brézet 63100 Clermont-Ferrand, France

Abstract. In this paper, we study the application of the box counting method (BCM) to estimate the fractal dimension of 3D plant foliage. We use artificial crowns with known theoretical fractal dimension to characterize the accuracy of the BCM and we extend the approach to 3D digitized plants. In particular, errors are experimentally characterized for the estimated values of the fractal dimension. Results show that, with careful protocols, the estimated values are quite accurate. Several limits of the BCM are also analyzed in this context. This analysis is used to introduce a new estimator, derived from the BCM estimator, whose behavior is characterized.

1 Introduction

Plant geometry is a key factor for modeling eco-physiological interaction of plant and the environment. These interactions may concern either the abiotic (resource capture, heat dissipation) or the biotic (disease propagation, insect movement) environment. Depending on applications, plant geometry has been abstracted in various ways [1] : simple volumic shapes (like ellipsoids, cones, or *big leaves* used in turbid medium approaches) or detailed models to render realistic trees. Global descriptions are simple and contain few parameters; however, they do not capture the irregular nature of plant shapes which severely limits the generalization capacity of the model. On the other hand, detailed descriptions tentatively address this problem but require over-parameterization of geometry, leading to non-parsimonious models. Characterizing the irregularity of plant shapes with a few parameters is thus a challenging problem.

ractal geometry was introduced as a new conceptual framework to analyze and model the irregular nature of irregular shapes [2]. This framework has been applied in different occasions to the modeling of plant structure. Generative approaches use fractal concepts to illustrate how intricate vegetal-like structures can be generated using parsimonious models [3,4,5]. Such models were used to generate artificial plants in modeling applications [6,7]. Fractal geometry was also used to analyze the irregularity of plants by determining their supposed fractal dimension. This parameter is of major importance in the study of irregularity: it characterizes the way plants physically occupy space. Most of these studies were carried out using the classical *box counting method* (BCM) [2] on woody structures, and especially on root systems [8,9,10]. This method consists of immersing the studied object in a grid with uniform cell size and studying the variation of the number of grid cells intercepted by the plant as the size of the cells decreases.

For practical reasons, in most works, fractal dimension is estimated from 2D photographs [11,12]. Unfortunately, such a technique always under-estimates the actual fractal dimension [13], and so is not accurate. Recently BCM was used on 3D digitized root systems [10]; however, the accuracy of the estimated values could not be evaluated. In this paper, we study the application of the BCM to both artificial and real 3D plant foliage. We use artificial crowns with known theoretical fractal dimensions to characterize the accuracy of BCM and we extend the approach to 3D digitized plants. The limits of BCM is then analyzed and discussed in this context.

2 Plant Databases

Nine 3D plants were included in the study. Four real trees were digitized in the field and five additional plants were generated from theoretical models. The geometric scenes representing the plant crowns were designed using the PlantGL library [14].

Digitized Plants. Four four-year old *Prunus Persica* (peach) trees were digitized [15], but due to the high number of leaves (∼14,000), digitizing at leaf scale was impossible. A magnetic digitizing device was therefore used to record the spatial co-ordinates of the bottom and top of each leafy shoot. In addition, thirty shoots were digitized at leaf scale in order to derive the leaf angle distribution, and allometric relationships between number of leaves, shoot leaf area and shoot length. Leaves of each shoot were then generated from those data and additional assumptions for the internode length and the distribution of leaf size within a shoot.

Theoretical Plants. Three fractal plants were generated from 3D *iterated function systems* (IFS) [4]. The generation process is illustrated in Fig. 2, and the finals artificial canopies are represented in Fig. 3. If the IFS satisfies the *open set condition* [16], the theoretical fractal dimension of the IFS attractor is the *autosimilarity dimension*,

$$D_a = \frac{\log n}{\log c}. \tag{1}$$

A classical 3D cantor dust [2] was also generated using an IFS ($n = 8, c = 3$). Each IFS was developed over 5 iterations. In addition to these self-similar plants

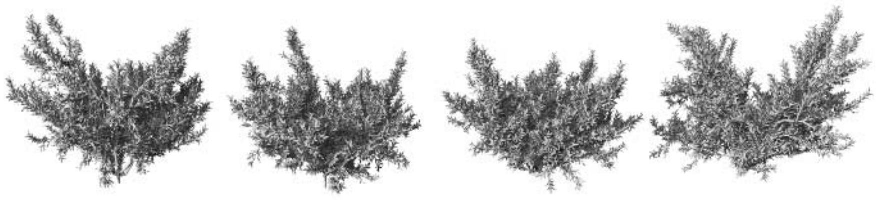


Fig. 1. Four four-year old peach trees (cv. August Red) were digitized in May 2001 in CTIFL Center, Nîmes, South of France, at current-year shoot scale, one month after bud break

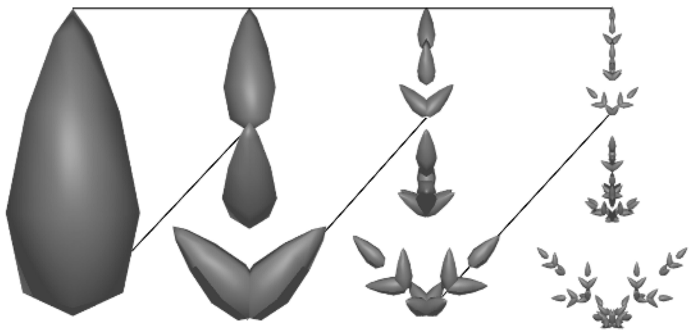


Fig. 2. Construction of an artificial crown. The initial object was a tapered ellipsoid and the IFS transformation was made of $n = 5$ duplications of a contracted object by a factor $c = 3$.

a stochastic 3D cantor dust was generated using a recursive algorithm derived from the method known as *curdling and random trema generation* [2,17]. Each iteration of the algorithm divides a given voxel into a set of subvoxels according to a specified subdivision factor. A fixed proportion of voxels eligible for the next iteration is chosen randomly from the subvoxels. At the end of the process, final voxels are considered to be leaves. The stochastic cantor dust is created by specifying a subdivision factor of 3 and $\frac{8}{27}$ as the proportion of chosen voxels for all 5 iteration levels. This object has the same theoretical dimension as the classical cantor dust.

3 Estimation of the Fractal Dimension Using the BCM

3.1 The Box Counting Method

The BCM has been extensively used to estimate fractal dimension of objects embedded in the plane. Its adaptation to 3D consists of building a sequence of 3D grids dividing space in homogeneous voxels of decreasing size δ and counting

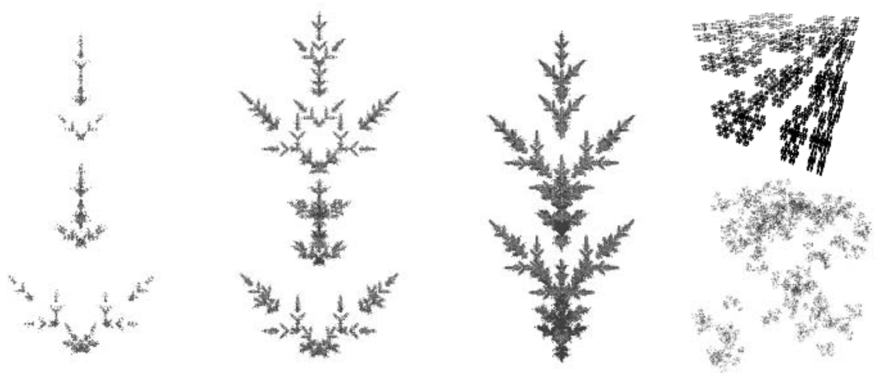


Fig. 3. From left to right, the three artificial canopies : AC1 ($n = 5, c = 3$), AC2 ($n = 7, c = 3$), AC3 ($n = 9, c = 3$), on the top, the cantor dust and on the bottom a stochastic cantor dust

the number N_δ of grid voxels intercepted by the studied object. The estimator of the fractal dimension of the object is defined as

$$D_b = \lim_{\delta \to 0} \frac{\log N_\delta}{\log \frac{1}{\delta}}. \tag{2}$$

To implement this estimator, we approximated all the geometric objects by triangular meshes. The intersection of each triangle with the grid voxels can then be computed in time proportional to the number of triangles in the mesh [18]. However, to decrease the overall complexity, we represent each triangle by a set of points [19]. The number of points used is chosen such as the distance between two points is small compared to the minimal voxel size. The intersection algorithm is thus reduced to checking whether a voxel contains at least one point. The grid sequence is obtained by dividing the original bounding-box size, δ_0, by a range of consecutive integers acting as subdivision factors. Thus the series of δ_n is a decreasing series formed by $\{\frac{\delta_0}{S_i}\}_{0 \le i < n}$ where S_i is the i^{th} subdivision factor. Each sub-grid fits perfectly in the original bounding-box. It is important to note that several factors may influence the accuracy of this method, *e.g.* the choice of a proper range of scales and the orientation and alignement of the grid [20,21]. In practice D_b is estimated as the slope of the regression line between $\log N_\delta$ and $\log \frac{1}{\delta}$.

3.2 Box Counting Method: Local Scale Variation Estimator

As pointed out in [22], a major problem of the BCM estimator is that the numbers of intercepted voxels at each scale are correlated positively, and the correlation structure is completely ignored in the estimation procedure. This violates the assumption of data independency used in regression analysis. The consequence is an underestimation of confidence interval associated with the estimated fractal dimension. To eliminate the correlation, we introduce a new estimator, namely local scale variation estimator (LSV), based on the relative increase of intercepted voxels against the relative decrease in scale. This estimator can be derived from the BCM estimator as follows. Assuming the power law is verified for each scale δ

$$N_\delta \propto (\frac{1}{\delta})^{D_b}, \tag{3}$$

the differential form of this equation leads to

$$\begin{aligned} d\log N_\delta &\propto d(D_b \log(\frac{1}{\delta})), \\ \frac{dN_\delta}{N_\delta} &\propto -D_b \frac{d\delta}{\delta} \end{aligned} \tag{4}$$

which gives a variational interpretation of the fractal dimension. D_b thus expresses the linear coefficient that corresponds to the ratio of new details due to a certain ratio of zoom in the structure. However, in this equation it is assumed that both dN and $d\delta \simeq 0$, which is not usually the case for the scales used in BCM, except at very small scales. It is possible to generalize this variational principle to non-infinitely small quantities. Let N_δ be the number of intercepted voxels at scale δ. We define $\Delta N_{\delta,\Delta\delta}$ as

$$\Delta N_{\delta,\Delta\delta} = N_{\delta+\Delta\delta} - N_\delta. \tag{5}$$

The relative increase in the number of boxes is denoted $\widetilde{N} = \frac{\Delta N_{\delta,\Delta\delta}}{N_\delta}$. Similarly, we denote $\widetilde{\delta} = \frac{\Delta\delta}{\delta}$ the relative increase of zoom when passing from cell size δ to $\delta + \Delta\delta$. Thus, assuming Equation 3 is still satisfied, we have

$$\widetilde{N} \propto \frac{(\delta + \Delta\delta)^{-D_b} - \delta^{-D_b}}{\delta^{-D_b}} = \left(1 + \widetilde{\delta}\right)^{-D_b} - 1, \tag{6}$$

which leads to a generalized form of Equation 4, where variations of N_δ and δ need not be infinitely small,

$$\log(1 + \widetilde{N}) \propto -D_b \log(1 + \widetilde{\delta}). \tag{7}$$

D_b can thus be estimated by performing a linear regression between $\log(1 + \widetilde{N})$ and $\log(1 + \widetilde{\delta})$.

4 Results

4.1 Number of Voxels as a Function of Scale

In general, we may expect that the number of intercepted voxels is a monotonously increasing function of scale. However this is not always the case due to a *quantization effect* which results from discrepancy between discretization with the 3D grid and space occupation of the plant at some scales. Fig. 4 contains plots of the number of voxels intercepted at the different scales for each object. The local variation of the curves comes from the fact that the number of intercepted voxels at one scale depends of the adjustment of the grid. Some shiftings, up to a factor δ in each direction, and reorientations of the grid may lead to overestimating the number of voxels at one scale, causing local variation of the curve. Thus, the discrete quantization of the 3D shape of the object into voxels introduces some fuzziness in its representation, depending on scale. It can be seen in Fig. 4 that the quantization effect is far more pronounced with the artificial crowns and Cantor dusts than the digitized peach trees. This difference is attributed to the less deterministically distributed foliage of the digitized trees.

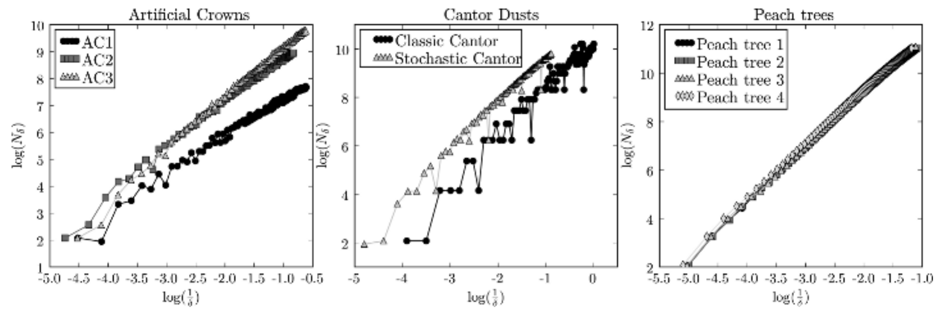


Fig. 4. The number of intercepted voxels as a function of the scale

4.2 Estimating Fractal Dimension from the BCM

Scale Range. When the grid voxel size is smaller than the leaf size, the evaluation of the dimension is modified by the dimension of the leaves surfaces. To avoid this effect, a minimum voxel size, δ_{min}, is determined such as $\delta_{min} \geq \sqrt{\mathcal{A}_l}$, where $\mathcal{A}_l$ is the mean leaf area. Since every voxel size δ_i is obtained from the bounding box size δ_0 as $\delta_i = \frac{\delta_0}{S_i}$, the minimum size must be $\delta_{min} = \frac{\delta_0}{S_{max}}$. Let $\mathcal{V}_{bb}$ be the bounding box volume. An uni-dimensional proportionality factor is defined by

$$S_{max} = \frac{\sqrt[3]{\mathcal{V}_{bb}}}{\sqrt{\mathcal{A}_l}}. \tag{8}$$

Setting S_{max} as the upper bound for the subdivision factors $\{S_i\}_{0 \leq i < n}$ guaranties that no voxel size will be smaller than a leaf size.

Grid Shifting. When the voxel size is close to the leaf size, the local adjustments of the grid may cause significant variations in the number of intercepted voxels, as discussed above. Practically, to limit the effect of this local variation due to grid shifting, a factor $\frac{S_{max}}{3}$ instead of S_{max} as the contraction limit was considered. This factor can be explained as follows. Let us consider a grid with voxels equal in size to the mean leaf size. Optimally a leaf will be included into a single voxel. All the possible shifting configurations of the grid may cause the leaf also be included in any of the twenty-six neighboring voxels. Considering voxels of bigger sizes with a factor 3 can be seen as including the twenty-seven possible small voxels into the same large one and so limits the errors found in finer grids. Of course, the optimal grid for one leaf will not be the optimal grid for all leaves; therefore, artifact effects of grid adjustment may persist. We experimentally observed that this persistence is limited (see Fig. 5).

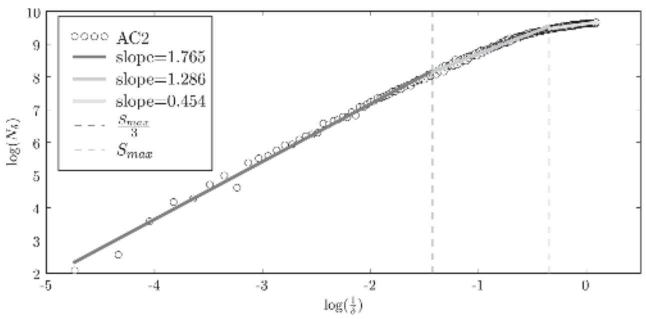


Fig. 5. Evolution of AC2 slope during BCM evaluation. The number of voxels intercepted at various scales for AC2 with the slopes highlighted. In the range $\left[0, \frac{S_{max}}{3}\right]$, the slope is primarily influenced by the structure of AC2 and the fractal dimension $D_b = 1.765$. In the range $\left[\frac{S_{max}}{3}, S_{max}\right]$, the slope is also partially influenced by the fractal dimension of individual leaves and is sensible to local variation due to grid adjustment. When this range is taken into account for the fractal dimension evaluation, D_b drops from 1.765 to 1.584. Finally for grids with voxel sizes smaller than S_{max}, the slope is directly related to individual leaf fractal dimension (0 in our representation since we use points). With a naive range of evaluation including all points, the fractal dimension drops to 1.172.

Orientation of the Grid. Optimal voxel coverage of the plant depends on the orientation of the grid relative to the plant. For this, we made a sensitivity analysis to evaluate how the estimated fractal dimension is affected by changes in the grid's orientation. A set of random grid orientations were selected and fractal dimension was estimated for each orientation. Table 1 gives the mean and variance of the estimated fractal dimension across orientations for all the considered plants. We can observe a low variability in the absolute values of the results: the standard deviations are inferior to one per cent of the mean values. From this, we conclude that the orientation of the grid has only limited effect on the BCM evaluation method.

Error Characterization. To characterize the error made during the estimation, a comparison with theoretical fractal dimension can be used. In the case of plants corresponding to IFS attractors, the theoretical fractal dimension, D, is known. But there is no such dimension for real plants; however, it has been shown that, when plant's topology is known, a faithful estimate of the plant fractal dimension can be obtained using the two-surface method [23]. This value will be used as reference value for the peach trees.

A classical *Student's t-test* on the computed D_b distributions shows that a significant bias in the BCM estimation exists. However, results reported in Table 1 (cols 3-6) show that this bias is less than 3.1% of the theoretical value for the studied canopies.

Table 1. Fractal dimension results for studied canopies and their properties. D_a is the reference (theoretical) value of the fractal dimension. For D_b estimation, $\overline{D_b}$ gives the mean estimated value and σ the standard deviation over all considered rotations. The minimum standard error r^2 over all rotations is shown. All results are obtained with $\frac{S_{max}}{3}$ as the upper limit.

		BCM D_b			Relative	LSV D_b					
Canopy	D_a	$\overline{D_b}$	σ	r^2	Bias	$\overline{D_b}$	σ	r^2	$\sqrt[3]{\mathcal{V}_{bb}}$	$\sqrt{\mathcal{A}_l}$	S_{max}
AC1	1.47	1.4889	0.0056	0.97	0.0128	1.8761	0.0457	0.33	1.83	0.0143	128
AC2	1.77	1.7305	0.0053	0.99	0.0223	1.9409	0.06	0.58	2.29	0.0143	160
AC3	2	1.97	0.0074	0.99	0.015	2.0705	0.0534	0.74	1.85	0.0143	129
Cantor	1.89	1.8835	0.0174	0.94	0.0034	2.2286	0.0852	0.09	0.99	0.0041	243
Stoc. Cantor	1.89	1.8896	0.0105	0.97	0.0002	2.1218	0.0933	0.17	2.43	0.01	243
Peach 1	2.33	2.3221	0.0043	0.99	0.0033	2.2832	0.0115	0.97	2.97	0.439	67
Peach 2	2.36	2.3516	0.0056	0.99	0.0035	2.3416	0.0117	0.97	2.97	0.459	64
Peach 3	2.38	2.307	0.0064	0.99	0.0306	2.3022	0.0195	0.97	3.04	0.0463	65
Peach 4	2.33	2.3218	0.0076	0.99	0.0035	2.3147	0.0175	0.98	2.61	0.0449	72

4.3 Estimating Fractal Dimension from the LSV Method

We use the LSV estimator of the box counting method, presented on section 3.2, on the theoretical and digitized plants. The $\widetilde{\delta}$ values were defined using couple of successive scales

$$\widetilde{\delta} = \frac{\delta_{i+1} - \delta_i}{\delta_i} = \frac{1}{\delta_i} \tag{9}$$

and $\widetilde{N}$ values from the corresponding N values. Since it is based on a local estimation, it is sensible to the local variation of the number of box as a function of scales introduced by the quantization effect. The local variations in this estimation are reflected in the variance and standard error of the computed fractal dimensions, giving a better estimation of the reliability of the results compared to the classical box counting method.

Experimentally, we observe that results on theoretical plants are very sensitive to quantization effect as shown by dispersion of the data in the Fig. 6 and the minimum standard error in Table 1 (cols 7-9). The minimum r^2 for the estimated dimensions on these objects are between 0.09 to 0.74. This effect is much less important on real plants; the minimum r^2 values are between 0.97 and 0.98. In this case, the results seems more relevant. The difference with theoretical values is small (less than 3.2%).

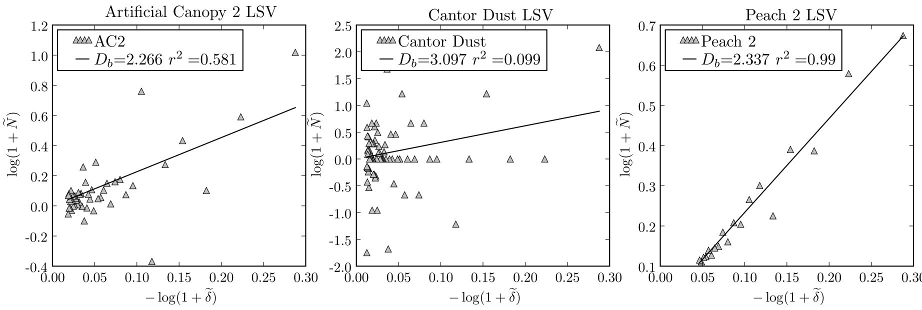


Fig. 6. Estimated fractal dimension with the LSV method for AC2, Cantor Dust and Peach 2. This new estimator is very sensitive to quantization effect leading to a dispersion of the measurements in AC2 and Cantor Dust. On the contrary the method gives an estimation of D close to that obtained with the two-surface method (i.e $D = 2.36$) for Peach 2 tree.

5 Conclusion

In this paper the accuracy of the BCM for evaluating the fractal dimension of 3D crowns was studied. Several factors that may influence this accuracy were examined and practical solutions proposed. In particular a proper voxel size limit is determined dependent on leaf sizes and the BCM bias was quantified. The problem of data dependency used during the regression analysis was discussed and a new estimator, LSV, that does not violate the independence assumption is described. The LSV estimator appears to be an interesting indicator to determine whether the quantization effect disturb the fractal dimension estimation. Eventually it has to be improved to support more robust evaluations.

References

1. Godin, C.: Representing and encoding plant architecture: A review. Annals of Forest Science **57** (2000) 413–438
2. Mandelbrot, B.B.: The fractal geometry of nature. Freeman (1983)
3. Smith, A.R.: Plants, fractals, and formal languages. In: Siggraph'84, Computer Graphics Proceedings. Volume 18., ACM Press (1984) 1–10
4. Barnsley, M.: Fractals Everywhere. Academic Press, Boston (1988)
5. Prusinkiewicz, P., Hanan, J.: Lindenmayer systems, fractals, and plants. Lecture Notes in Biomathematics **75** (1989)
6. Chen, S., Ceulemans, R., Impens, I.: A fractal-based *Populus* canopy structure model for the calculation of light interception. Forest Ecology and Management **69**(1-3) (1994) 97–110
7. Prusinkiewicz, P., Mundermann, L., Karwowski, R., Lane, B.: The use of positional information in the modeling of plants. In: Siggraph'01, Computer Graphics Proceedings, New York, NY, USA, ACM Press (2001) 289–300
8. Fitter, A.H.: An architectural approach to the comparative ecology of plant root systems. New Phytologist **106**(1) (1987) 61–77

9. Eshel, A.: On the fractal dimensions of a root system. Plant, Cell & Environment **21**(2) (1998) 247+
10. Oppelt, A.L., Kurth, W., Dzierzon, H., Jentschke, G., Godbold, D.L.: Structure and fractal dimensions of root systems of four co-occurring fruit tree species from *Botswana*. Annals of Forest Science **57** (2000) 463–475
11. Morse, D.R., Lawton, J.H., Dodson, M.M., Williamson, M.H.: Fractal dimension of vegetation and the distribution of arthropod body lengths. Nature **314**(6013) (1985) 731–733
12. Critten, D.L.: Fractal dimension relationships and values associated with certain plant canopies. Journal of Agricultural Engineering Research **67**(1) (1997) 61–72
13. Falconer, K.: Fractal geometry : mathematical foundation and applications. John Wiley and Sons (1990)
14. Boudon, F., Pradal, C., Nouguier, C., Godin, C.: Geom module manual: I user guide. Technical Report 3, CIRAD (2001)
15. Sonohat, G., Sinoquet, H., Kulandaivelu, V., Combes, D., Lescourret, F.: Three-dimensional reconstruction of partially 3d-digitized peach tree canopies. Tree Physiol **26**(3) (2006) 337–351
16. Falconer, K.: Techniques in fractal geometry. John Wiley and Sons (1997)
17. Plotnick, R.E., Gardner, R.H., O'Neill, R.V.: Lacunarity indices as measures of landscape texture. Landscape Ecology **8** (1993) 201–211
18. Andres, E., Nehlig, P., Franon, J.: Supercover of straight lines, planes and triangles. In: Proceedings of DGCI '97, London, UK, Springer-Verlag (1997) 243–254
19. Pfister, H., Zwicker, W., Baar, J.v., Gross, M.: Surfels: surface elements as rendering primitives. In: Siggraph'00, Computer Graphics Proceedings, Los angeles, ACM Press (2000) 335–342
20. Foroutan-Pour, K., Dutilleul, P., Smith, D.L.: Advances in the implementation of the box-counting method of fractal dimension estimation. Applied Mathematics and Computation **105**(2) (1999) 195–210
21. Halley, J.M., Hartley, S., Kallimanis, A.S., Kunin, W.E., Lennon, J.J., Sgardelis, S.P.: Uses and abuses of fractal methodology in ecology. Ecology Letters **7** (2004) 254–271
22. Reeve, R.: A warning about standard errors when estimating the fractal dimension. Comput. Geosci. **18**(1) (1992) 89–91
23. Boudon, F., Godin, C., Pradal, P., Puech, O., Sinoquet, H.: Estimating the fractal dimension of plants using the two-surface method. an analysis based on 3d-digitized tree foliage. Fractals **14**(3) (2006)

3D Surface Reconstruction and Registration for Image Guided Medialization Laryngoplasty

Ge Jin[1], Sang-Joon Lee[1], James K. Hahn[1], Steven Bielamowicz[2], Rajat Mittal[3], and Raymond Walsh[4]

[1] Department of Computer Science
[2] Division of Otolaryngology, School of Medicine
[3] Department of Mechanical and Aerospace Engineering
[4] Department of Anatomy and Cell Biology, School of Medicine
[5] The George Washington University, Washington DC, USA
{jinge, bigdolph, hahn}@gwu.edu, sbielamowicz@mfa.gwu.edu, mittal@gwu.edu, anarjw@gwumc.edu

Abstract. The purpose of our project is to develop an image guided system for the medialization laryngoplasty. One of the fundamental challenges in our system is to accurately register the preoperative 3D CT data to the intraoperative 3D surfaces of the patient. In this paper, we will present a combined surface and fiducial based registration method to register the preoperative 3D CT data to the intraoperative surface of larynx. To accurately model the exposed surface area, an active illumination based stereo vision technique is used for the surface reconstruction. To register the point clouds from the intraoperative stage to the preoperative 3D CT data, a shape priori based ICP method is proposed to quickly register the two surfaces. The proposed approach is capable of tracking the fiducial markers and reconstructing the surface of larynx with no damage to the anatomical structure. Although, the proposed method is specifically designed for the image guided laryngoplasty, it can be applied to other image guided surgical areas. We used off-the-shelf digital cameras, LCD projector and rapid 3D prototyper to develop our experimental system. The final RMS error in the registration is less than 1mm.

Keywords: Image Guided Surgery, 3D Reconstruction, Registration.

1 Introduction

It is estimated that 7.5 million people in the United States have a voice disorder, and about 1/3 of new patients with voice disorders are diagnosed with vocal fold paresis or paralysis. Vocal cord paralysis and paresis are debilitating conditions leading to difficulty with voice production. The alterations in voice production are usually severe enough to impede the individual's ability to work and to conduct normal social interactions. Medialization laryngoplasty is a surgical procedure designed to restore the voice in patients by implanting a uniquely configured structural support lateral to the paretic vocal fold through a window cut in the thyroid cartilage of the larynx. Currently, the surgeon relies on experience and intuition to place the implant in the desired location, therefore it is subject to a significant level of uncertainty. Window placement errors of up to 5mm in the vertical dimension are common in patients admitted for revision surgery. The failure rate of this procedure is as high as 24% even for experienced surgeons [1]. An intraoperative image guided system will help the surgeon to accurately place the implant at the desired location.

The image guided technology has been successfully applied to various medical domains. However, to our knowledge, image guided techniques have not been applied to the medialization laryngoplasty. The biggest obstacles come from (1) registering the geometry of the delicate anatomy of thyroid cartilage during the surgery to the preoperative 3D CT data (2) introducing minimal intrusion or modifications to the current surgical practices and (3) implementing with only a moderate increase in the additional equipment. In this paper, we will concentrate on the registration of preoperative 3D CT data to the intraoperative 3D surfaces of thyroid cartilage.

Our proposed image guided system will use the anatomical and geometric landmarks and points to register intraoperative 3D surface of thyroid cartilage to the preoperative 3D radiological data. The proposed approach has three phases. First, the laryngeal cartilage surface is segmented out from the preoperative 3D CT data. Second, the surface of the exposed laryngeal cartilage during the surgery is reconstructed intraoperatively using stereo vision and structured light based surface scanning. The surgical area has non-uniform color and textures, so we take one full-lit image and non-lit image to distinguish the shadow from the light receiving areas and calculate the illumination change map. Third, the two geometries are registered using shape priori based ICP matching. Currently the proposed technqiue has only been applied in a laboratory environment on phantom models. The proposed approach has several advantages over alternative approaches: the combination of stereo vision and structured light surface scanning is capable of tracking the fiducial markers, reconstructing the surface of laryngeal cartilage and matching the preoperative and postoperative surfaces for registration purposes. The computer vision based approach can be applied to delicate areas like laryngeal cartilage with no danger of causing physical damage.

2 Background

Registration in image guided procedures can be classified into three categories based on the fiducial markers: extrinsic invasive (bone affixed markers)[2][3][4], extrinsic noninvasive (skin affixed markers)[5] and intrinsic markers [6]. In the case of laryngoplasty, the bone fixed fiducial markers would make potentially damage to the thin laryngeal cartilage. While, the skin affixed markers will move significantly relative to the laryngeal cartilage. Intraoperative medical imaging system can be used for the multi-modal image registration in image guided surgery [7][8]. However, for the medialization laryngoplasty, this will modify the current surgical procedure and increase the medical cost by introducing additional medical equipment. So, in our system, we will use the intrinsic markers for the registration.

The structured light based surface reconstruction system can be classified into three categories: time-multiplexing, spatial neighborhood and direct coding. Spatial neighborhood [9][10] and direct coding [11] methods are relatively fast and capable of measuring dynamic surfaces. However, the bandwidth of projector and quantization error introduced by the CCD camera will make the color and neighborhood based methods less accurate than time multiplexing methods. Time-multiplexing is a way to encode the pixel information in the temporal domain. Posdamer and Altschuler [12] first proposed a 3D surface measurement method with binary coded light pattern. Inokuchi [13] further improved the coding scheme using gray code to make the code-word robust to the noise. Recently, Gühring [14] combined the gray code light pattern and line shifting to reconstruct highly accurate 3D surface model. For our experimental framework, since the primary goal is to reconstruct accurate 3D surface for registration, we used sub-pixel accuracy line shifting method to reconstruct the 3D surfaces.

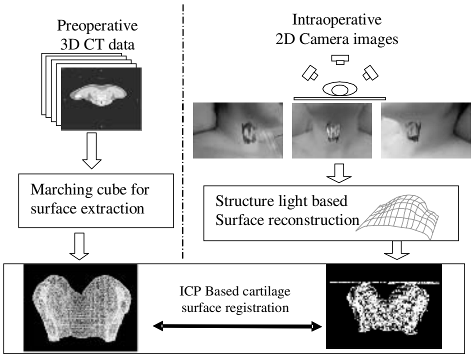
The global alignment of multiple 3D point sets or surfaces has been well studied in the field of 3D model acquisition area. ICP (Iterative Closest Point) algorithm was introduced to geometrically align two similar geometric models [15][16]. A new geometric transform matrix is calculated by minimizing the MSE (Mean Square Error) between the closest point pairs. Horn [17] described a closed form solution for the quaternion calculation from the closest point pairs. Kd-tree [15], and approximated kd-tree [18] is used to accelerate the closest point searching process. Recently, sub-sampling scheme from the geometric data, closest point searching method, rejection of outliers and error minimization method are used to compare various ICP algorithms [19]. In our case, the number of points from the preoperative CT and structured light based surface scanning are relatively small (about 3000 points), so we used all the point samples during the ICP matching process. For the closest point searching, a balanced kd-tree is used to accelerate the searching speed. We used the closed form solution from [17] to calculate the unit quaternion rotation vector, and rejected the outliers from sample space if the closest distance is longer than 2 times of mean closest distance. For our case, the shape features of the laryngeal cartilage will be a good candidate for fast initial pose estimation. We used two crossing planes to calculate the initial pose for fast shape matching.

3 Image Guided Medialization Laryngoplasty

The work flow of our surface registration process is shown in Figure 1 Left. There are three major steps: surface extraction from preoperative CT data, structured light based intraoperative surface reconstruction and ICP based point clouds registration.

3.1 Medialization Laryngoplasty

The medialization laryngoplasty (Figure 1. Right) procedure is the thyroplasty procedure, which is aimed at medializing the membraneous aspects of the vocal fold. A thyroplasty implant is a patient-specific device that must be properly



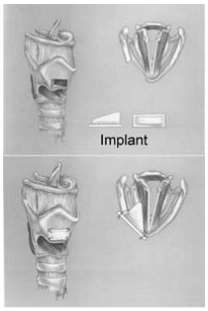


Fig. 1. Left:Work flow of surface registration, Right: Medialization laryngoplasty

aligned in reference to the underlying vocal fold and have a size and shape such that it medializes the vocal fold and alters the vibratory characteristics of the vocal fold to a state that most closely resembles that of the uninjuried vocal cord.

3.2 Surface Extraction from CT Data and Phantom Model Construction

We used visible human CT data set from NIH for our experiment. The thyroid cartilage surface is extracted using marching cube algorithm [20]. The extracted triangular mesh is rendered in wire frame, flat shading and texture mapping (Figure 2. Left). The extracted 3D surface model is converted to a solid CAD

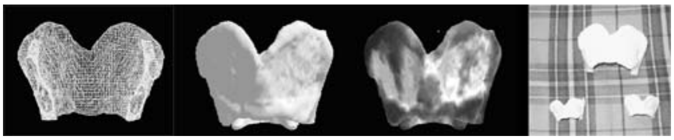


Fig. 2. Left: Iso-surface extraction from the CT data, Right: Phantom model

model and sent to the 3D prototyping device (Stratasys FDM 3000). The prototyper is capable of constructing a 3D phantom model with the accuracy of 0.1mm (Figure 2. Right).

3.3 Structured Light Based Intraoperative Surface Reconstruction

In the surgical environment, the area of scanning has non-uniform color and texture. Threshold based image segmentation could not provide accurate structured light pattern. Similar with photometric calibration of projector and camera, one full-lit image and non-lit image are used to distinguish the shadow from the light receiving areas. Another difficulty in reconstructing laryngeal cartilage surface is the small size of anatomical structure. Usually, the structured light based surface scanning is applied to rather big structures like: human faces, statues and so on. The fully exposed larynx is about 90x50x50 mm. If the distance from camera and laryngeal cartilage is larger than certain distance, a regular camera with standard resolution (640X480) could not provide enough resolution. Since the camera should not disturb the surgical procedures, there is a minimum distance requirement for the surgical environment. With this restriction in mind, a higher resolution camera is required to provide enough accuracy for surface reconstruction.

Structured light based surface reconstruction requires light projection device (LCD projector) and one or more cameras. In our case, we used LCD projector with two cameras. Since the camera to camera calibration has higher accuracy than camera to projector calibration, we only calibrated the camera pairs and used the LCD for illumination purpose. For the camera calibration, we used the planar homography based camera calibration method from [21].

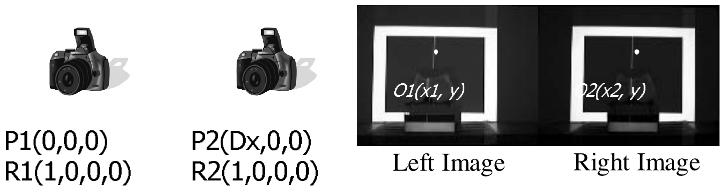


Fig. 3. Camera parameter after rectification and rectified images

After calibration, the images from two cameras are rectified to align the horizontal scan lines. After rectification, the searching of pixel correspondence has been reduced to one dimension. Furthermore, the camera internal and external parameters are simplified. In figure 3, the P_1, P_2 is the camera position vector and the R_1,R_2 is the cam-era rotation matrix represented by the quaternion. In equation 1, the M_1, M_2 is the pinhole camera projection matrix. If we find the pixel correspondence (O_1 and O_2) in left and right images, we can calculate the real 3D position of the pixel in camera coordinate system by solving the linear equations shown on (1).

$$M_1 = \begin{bmatrix} f\ 0\ C_{x1} \\ 0\ f\ C_y \\ 0\ 0\ 1 \end{bmatrix};\ M_2 = \begin{bmatrix} f\ 0\ C_{x2} \\ 0\ f\ C_y \\ 0\ 0\ 1 \end{bmatrix};\ \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = M_1^{-1} \begin{bmatrix} x_1 \\ y \\ 1 \end{bmatrix} = M_2^{-1} \begin{bmatrix} x_2 \\ y \\ 1 \end{bmatrix} \tag{1}$$

From the above equation, we can easily notice that the sub-pixel accuracy in pixel correspondence is the most critical issue in 3D reconstruction. If we only have the pixel level accuracy, the recovered depth value will not be continuous. So, we experimented with sub-pixel accuracy line shifting method to reconstruct the surface of thyroid cartilage phantom model. First, we searched for the peak intensity along the horizontal scan line. Then, we used the 7 nearby pixels for the sub pixel peak detection and calculated the 2nd order derivative for 5 pixels around the detected peak. The sub-pixel intensity peak is calculated with zero-crossing of 2nd order derivatives. In the sub-pixel accuracy peak detection, the camera shutter speed, film sensitivity and projector focus simultaneously affect the peak detection result. The preliminary experiment has indicated that: the focus of the beam projector should focus on the laryngeal cartilage surface to provide maximum intensity variation and the camera shutter speed needs to be adjusted to capture the sub-pixel illumination change. If the image is over-exposed, the peak of the light strip will spread over several pixels and as a result the detected peak is not accurate.

3.4 ICP Based Point Clouds Registration

To register the 3D surface from preoperative CT data and the point clouds from the structured light based surface reconstruction, we need to preprocess the 3D surface from CT. The point clouds from the computer vision are only the front side of the thyroid cartilage. Therefore, we need to remove the back facing polygons from the preoperative CT surface so that the back facing polygons do not affect the registration result. We used the surface normal to separate the front facing and back facing polygons. In order to reduce the searching time for the closest point matching, we used balanced k-d tree. A kd-tree is a space-partitioning data structure for organizing points in a k-dimensional space. It uses splitting planes that are perpendicular to one of the coordinate system axes (Figure 4. right).

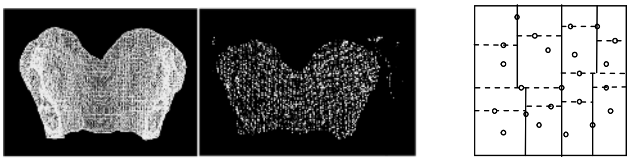


Fig. 4. Left: 3D model from CT and from structured light Right: 2D kd-tree

We used the point to point euclidian distance as our closest point matching criteria. After the calculation of closest point, we rejected the outliers from sample space if the closest distance is longer than 2 times of mean closest distance. The minimization of mean square error is only considered on inliers.

Suppose M and D are 3D point sets from preoperative and intraoprative stages, the goal of ICP algorithm is to find the optimal rotation and translation that minimize equation $E(R,T) = \sum_{i \in M} \sum_{j \in D} \| m_i - (R \cdot d_j + T) \|$. We used unit quaternion Q(q_0,q_1,q_2,q_3) to represent the rotation matrix R.

$$R = \begin{bmatrix} q_0^2+q_1^2+q_2^2+q_3^2 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & q_0^2+q_2^2-q_1^2-q_3^2 & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & q_0^2+q_3^2-q_1^2-q_2^2 \end{bmatrix} \quad (2)$$

The closed form solution from [17]'s work is used to calculate the quaternion vector. To determine the rotation vector, we first subtract center of mass position from each point clouds set. A covariance matrix N is calculated using the equation 3 where $S_{xx} = \sum\sum m^*_{ix} * d^*_{jx}$. The new quaternion vector Q is the eigenvector of largest positive eigen value of N.

$$N = \begin{bmatrix} S_{xx}+S_{yy}+S_{zz} & S_{yz}-S_{zy} & S_{zx}-S_{xz} & S_{xy}-S_{yx} \\ S_{yz}-S_{zy} & S_{xx}-S_{yy}-S_{zz} & S_{xy}+S_{yx} & S_{zx}+S_{xz} \\ S_{zx}-S_{xz} & S_{xy}+S_{yx} & S_{yy}-S_{xx}-S_{zz} & S_{yz}+S_{zy} \\ S_{xy}-S_{yx} & S_{zx}+S_{xz} & S_{yz}+S_{zy} & S_{zz}-S_{yy}-S_{xx} \end{bmatrix} \quad (3)$$

The original ICP algorithm calculates the translation vector using the difference in the center of mass point. This is correct when the center of mass points in preoperative and intraoperative surfaces are close. But, in our case, the surface points from the preoperative CT consist of points that are not exposed to the camera. Furthermore, the structured light based reconstruction stage also consists of noise points. So we separated the translation calculation from rotation calculation stage. For the translation, we used the summed average of displacement vector of matched closest point pairs. Suppose [X$_{M1}$,Y$_{M1}$,Z$_{M1}$] and [X$_{M2}$,Y$_{M2}$,Z$_{M2}$] is closest matching point pair from CT and structured light based reconstruction, the new translation vector is calculated using $[T_x, T_y, T_z] = [\sum_n (X_{M1} - X_{M2})/n, \sum_n (Y_{M1} - Y_{M2})/n, \sum_n (Z_{M1} - Z_{M2})/n]$.

The initial pose estimation will greatly affect the convergence speed and the correctness of the final result. Unlike original ICP based shape matching, for the medical image registration, the ground truth target mesh is known. The shape features of the laryngeal cartilage will be a good candidate for fast initial pose estimation. One important observation is that the laryngeal cartilage surface can be approximated by two crossing planes (Figure.5). Point to plane distance $(\frac{|ax_0+by_0+cz_0+d|}{\sqrt{a^2+b^2+c^2}})$is used to estimate the plane equation $(ax + by + cz + d = 0)$. Minimizing the sum of squared distance from point to plane will provide a plane equation that best fit the point clouds. The center of mass of point clouds is projected to the plane to provide the unique matching point on the plane. The SVD based closed form solution is used to approximate the plane equation. The plane equation is the vector associated with smallest singular value (Equation 4). Geometric description based on initial shape approximation will provide a close initial pose estimation for the ICP method.

$$SS\_Dist = \sum_{v \in M} \frac{(ax+by+cz+d)^2}{a^2+b^2+c^2}; \; SVD: \; D = \frac{1}{N} \sum_{v \in M} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}^T \quad (4)$$

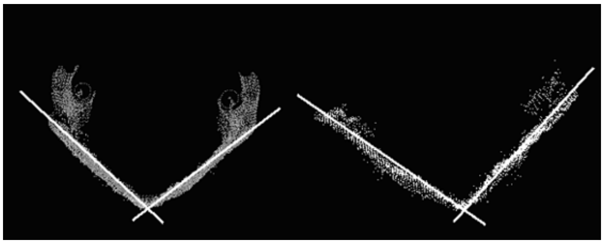


Fig. 5. Approximation of the larynx with two crossing planes

4 Experiment and Result

We used Intel Xeon 3.2GHz Workstation with 2GB memory for our experiment. For the structured light based surface reconstruction, we have experimented with two Logitec Quickcam cameras, Nikon D70s digital cameras and LCD projector. The surface reconstruction result with sub-pixel accuracy line shifting is show on figure 6 and figure 7 left.

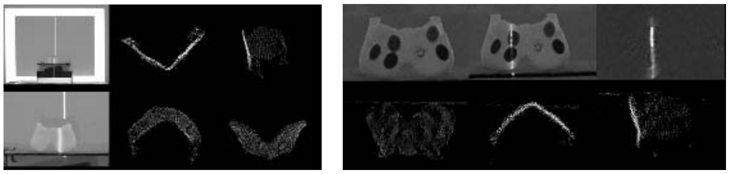


Fig. 6. Surface reconstruction result for phantom model

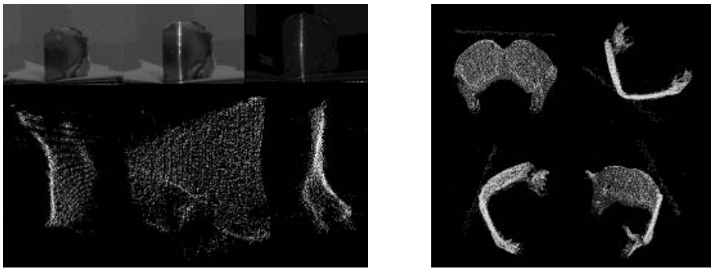


Fig. 7. Left: Animal bone surface reconstruction Right: Shape priori based ICP matching

To mimic the real situation, color dotted phantom model and animal bone are used for the experiment. The illumination change value is calculated by dividing the illuminated image with non-lit image. In ICP based point clouds registration, the computation time for kd-tree construction is 94 ms. Shape priori based ICP matching takes 515 ms to match the two point clouds with RMS error 0.9mm.

The original ICP method with the same RMS error takes 4 sec. The final mean square error in two matched point clouds is 0.899mm and the registration result is shown on figure 7 right.

5 Conclusions and Future Works

In this paper, we proposed an image guided system for the medialization laryngoplasty. To our knowledge, this is the first attempt to apply image-guided techniques to the medialization laryngoplasty. Due to the delicate nature of thyroid cartilage surface, we could not directly use the fiducial marker based optical tracking system for the image registration. Instead, we introduced a structured light based stereo vision system that could be used for 3D surface reconstruction and feature tracking. We used the sub-pixel accuracy line shifting for the 3D reconstruction. To mimic the real situation, color dotted phantom model and animal bone is used for experiment. Instead of using the absolute intensity value, the illumination change map is used for light peak detection. To match the 3D surface from preoperative CT and the point clouds from structured light based reconstruction, we proposed a shape priori based initial pose estimation combined with the ICP algorithm to register two sets of point clouds. The mean square error of ICP based registration is less than 1.0mm. Our experimental framework can be applied to other image guided applications. For the future work, we will use the registration result and the projective texture mapping techniques to render the preoperative thyroid cartilage surface and visualize the important anatomical structures (vocal fold and airway lumens) beneath the thyroid cartilage surface. This work is supported by a grant from the National Institute of Health (No. R01-DC007125-0181) for developing computer-based tools for medialization laryngoplasty.

References

1. T. D. Anderson, J. R. Spiegel, R. T. Sataloff: Thyroplasty revisions: frequency and predictive factors. Journal of Voice, Vol. 17(3): (2003) 442–448
2. P.J Kelly: Computer-assisted stereotaxis: new approaches for the management of intracranial intra-axial tumors. Neurology, Vol. 36(4): (1986) 535–541
3. K. P. Gall, L. J. Verhey, Wagner M: Computer-assisted positioning of radiotherapy patients using implanted radioopaque fiducials. Medical physics. Vol. 20(4): (1993) 1153–1159
4. Maurer CR Jr, Fitzpatrick JM, Wang MY, Galloway RL Jr, Maciunas RJ, Allen GS: Registration of head volume images using implantable fiducial markers. IEEE Trans Med Imaging Vol. 16(4): (1997) 447–462.
5. Darabi K, Grunert P, Perneczky A: Accuracy of intraoperative navigation using skin markers. In: CARS 1997. Berlin, Springer-Verlag, (1997) 920–924.
6. M. I. Miga, T. K. Sinha, D. M. Cash, R. L. Galloway, and R. J. Weil: Cortical Surface Registration for Image-Guided Neurosurgery Using Laser-Range Scanning IEEE Transaction of Medical Imaging, Vol. 22(8): (2003) 973–985

7. Wells, W. M., Viola, P., Atsumi, H., Nakajima, S., Kikinis, R.: Multi-modal volume registration by maximization of mutual information. Medical Image Analysis, Vol. 1(1): (1996) 35–51
8. H. Livyatan, Z. Yaniv, L. Joskowicz: Gradient-Based 2D/3D Rigid Registration of Fluoroscopic X-ray to CT. IEEE Trans. Med. Imaging Vol. 22(11): (2003) 1395–1406
9. K. L. Boyer, A. C. Kak: Color-encoded structured light for rapid active ranging. IEEE Trans. Pattern Anal. Mach. Intell., Vol. 9(1): 1987 14–28
10. J. Salvi, J. Batlle, E. Mouaddib: A robust-coded pattern projection for dynamic 3D scene measurement. Pattern Recogn. Lett., Vol. 19(11): (1998) 1055–1065
11. C. J. Davies, M. S. Nixon: A hough transform for detecting the location and orientation of 3-dimensional surfaces via color encoded spots: IEEE Trans. Systems, Man and Cybernetics, Vol. 28(1): (1998) 90–95
12. J. L. Posdamer, M. D. Altschuler: Surface measurement by space-encoded projected beam systems. Computer Graphics and Image Processing Vol. 18(1): (1982) 1–17
13. S. Inokuchi, K. Sato, F. Matsuda: Range imaging system for 3-D object recognition. In: Proc. of the International Conference on Pattern Recognition, (1984) 806–808
14. J. Gühring: Dense 3-d surface acquisition by structured light using off-the-shelf components. In SPIE Photonics West Videometrics, Vol. 4309: (2001) 220–231
15. Besl P. J., Mckay N. D.: A method for registration of 3d shapes. IEEE Trans. Pattern Anal. Mach. Intell., Vol. 14(2): (1992) 239–256
16. Chen Y., Medioni G.: Object modelling by registration of multiple range images. Image Vision Comput., Vol. 10(3): (1992) 145–155
17. B.K.P. Horn: Closed-form solution of absolute orientation using unit quaternions. Journal of the Optical Society of America, Vol. 4(4): (1987) 629–642
18. Greenspan, M., Yurick, M.: Approximate k-d tree search for efficient ICP. In Proc. 3D Digital Imaging and Modeling, (2003)442–448
19. S. Rusinkiewicz, M. Levoy: Efficient Variants of the ICP Algorithm. In 3rd Int. Conf. on 3D Digital Imaging and Modeling, (2001)
20. W. E. Lorensen and H. E. Cline: Marching cubes: a high resolution 3D surface construction algorithm. In Proc. SIGGRAPH 1987 Vol. 21(4): (1987) 163–170
21. Zhengyou Zhang: Flexible camera calibration by viewing a plane from unknown orientations. In 7th IEEE Int. Conf. on Computer Vision, (1999) 666–673

Vision-Based User Interfaces for Health Applications: A Survey

Alexandra Branzan Albu

Dept. of Electrical and Computer Engineering, University of Victoria (BC), Canada
aalbu@ece.uvic.ca

Abstract. This paper proposes a survey of vision-based human computer interfaces for several key-fields in health care: data visualization for image-guided diagnosis, image-guided therapy planning and surgery, the operating room, assistance to motor-impaired patients, and monitoring and support of elderly. The emphasis is on the contribution of the underlying computer vision techniques to the usability and usefullness of interfaces for each specific domain. *It is also shown that end-user requirements have a significant impact on the algorithmic design of the computer vision techniques embedded in the interfaces.*

1 Introduction

The field of computer vision focuses on the development and implementation of algorithms which allow computers to "understand" image and video data at various levels depending on the task at hand. Task-oriented image "understanding" may offer assistance to human perception, cognition and decision-making, such as in computer-aided diagnosis systems, or may enable more natural ways for human-computer interaction (HCI) in perceptual interfaces and in pervasive computing systems. Such vision-based technologies find promising applications in several areas of health care, including but not limited to image-based diagnosis and therapy planning, minimally invasive surgery, assistance and support for people with disabilities and elderly.

However, HCI design for medical applications is a difficult problem. Gosbee and Ritchie [1] built a hierarchical model of clinician acceptance of technology in an attempt to identify why physicians and other care providers are reluctant to introduce new HCI technology in their daily work routine. Considering this model, it is expected that a successful integration of computer vision algorithms in health-related HCI should consider both user-centered and task-based design paradigms. In return, these paradigms influence the basic assumptions as well as the algorithmic development of computer vision techniques.

This paper presents a survey of vision-based HCIs for several key-fields in health care: data visualization for image-guided diagnosis, therapy planning and surgery (section 2), the operating room (section 3), assistance to motor-impaired patients (section 4), and monitoring and support of elderly (section 5). The emphasis is on the contribution of the embedded computer vision techniques to the usability and usefullness of interfaces for each specific domain. Section 6 presents a summary of our survey and draws conclusions.

2 Vision-Based Interfaces for Enhanced Data Visualization

Computer vision algorithms for medical image analysis form four main groups with different scopes, as follows: a) filtering for image enhancement; b) segmentation for object delineation; c) analysis for feature extraction; d) image registration for multi-modal data fusion. Recently, applications for advanced medical data visualization such as virtual or augmented reality systems integrate computer vision with computer graphics. While computer vision deals with extracting relevant information from images (e.g. object boundaries, interstructural distances), computer graphics focuses on image synthesis for creating realistic and manipulable 3D objects representing anatomical structures of interest. Computer vision and computer graphics techniques are strongly interconnected in user interfaces (UI) for data visualization.

HCI for computer-aided diagnosis and therapy planning in clinical environments. In this context, a particular attention is to be directed towards the algorithmic design of segmentation techniques in order to meet end-user requirements collected from radiologists. Yet, relatively recent surveys on interactive and fully automated methods for medical image segmentation [2, 3] do not discuss the relevance of end-user requirements for segmentation design. One may reach the surprising conclusion that the development of most segmentation methods is not driven by end-user requirements; most developers focus on improving accuracy, precision and computational speed, as well as on reducing the amount of manual interaction. Widely recognized validation protocols for medical image segmentation [4, 5] perform a strict scientific comparison of the performance of automatic/semi-automatic methods against manual expert segmentations. Few methods search clinical feedback and conduct a usability study. Such a method is proposed by O'Donnell et al [6], and consists in a user-steered segmentation algorithm based on the interactive livewire paradigm [7]. Usability and usefullness are investigated using feedback from radiologists on the quality of segmentation and on the learnability of the UI.

Elliott et al [8] conducted one of the first systematic clinical studies for comparing the usefulness of two interactive segmentation methods [9] embedded in a graphical user interface for radiation therapy planning. Segmentation was task-oriented and aimed at outlining fast the target volume and organs at risk for 3D radiation treatment planning. The strategy in [8] for integrating segmentation algorithms into a HCI for radiologists' use was to make sure that the user's clinical knowledge is efficiently complemented by the segmentation algorithm. According to [8], "*…automated image segmentation is in any case not wanted by the users. What they do want is a fast system (i.e. one that is faster than manual segmentation) in which the user has complete control over the results*." Moreover, since radiologists tend to 'think in slices', 2D user interaction on a slice-by-slice basis was preferred over 3D interactive segmentation.

Shifting from slice-based 2D to full 3D user interaction is still an open question for diagnosis-oriented interfaces [10], which impacts not only on computer graphics, but also on computer vision techniques. Possible solutions for speeding up this shift focus on establishing new visualization standards for 3D image interpretation by radiologists [11], and on integrating 3D in the medical training curriculum [12].

Both 2D and 3D visualization paradigms were considered in the design of the 3D Slicer software and visualization platform [13], which offers a variety of tools for slice editing and interactive segmentation. The 3D Slicer interface is not intended for clinical use, but its design considers a large diversity of user profiles including research-oriented physicians.

Research-oriented graphical interfaces. To date, there is a rich literature on algorithms for 2D and 3D medical image filtering, registration, segmentation and analysis [14] standing proof for the significant advances in image interpretation made in the last decade. To increase the visibility of such new methods, an extensive open-source library called the Insight Toolkit (ITK) has been built [15]. The algorithms available within ITK are programmed into C++ classes and may be considered as building blocks for a variety of task-oriented applications in medical imaging. While ITK is a valuable resource to the Computer Vision research community, it is of limited use to end-users such as radiologists and surgeons, since it does not provide a graphical interface for image visualization. The ANALYZE software system [16] is complementary to ITK, since it provides tools for interactive visualization. Augustine et al [17] proposed an integration of ITK and ANALYZE, which results in a user-friendly graphical interface for research-oriented end-users with little programming skills. The main modules of this tabbed interface offer various tools for filtering, registration and segmentation including control windows for parametric adjustment. A fast display updating mechanism provides visual information about the evolution of image processing algorithms. User error is minimized by constraining the sequential order of typical operations (i.e. noise reduction, edge detection, segmentation, analysis). Other integration efforts focused on adding dynamic visualization functionalities to ITK were reported by Rexilius et al [18] and Hansen et al [19].

Towards HCI design for collaborative and remote image analysis. Computer supported collaborative work is helpful in medical applications requiring the expertise of more than one physician or dedicated to clinical training. A task-oriented, collaborative interface for the visualization and analysis of fetal 3D ultrasound is proposed by Alberola-Lopez et al [20]. Their interface allows for session-based work, therefore minimizing errors caused by user fatigue. In [20], segmentation and/or 3D data manipulation are performed by one user who has acquired control through a token-grabbing paradigm; the other users are ‘listeners’ until the token is released. Text-based information exchange is asynchronuous, therefore feedback or discussions on a specific graphical model can take place at any time. The token-grabbing paradigm was also implemented in the Group-Slicer [21], the collaborative extension of 3D Slicer.

HCI for enhanced visualization during image-guided surgery. During open surgery, the surgeon has direct visual access only to exposed surfaces; the limitations are even more severe for minimally invasive surgery. The main consequence of limited surgical visualization is the non-accuracy of the pre-operative and intraoperative geometric localization of the targeted lesion. Three basic user requirements for the design of intraoperative image guidance interfaces are defined by McInerney and Roberts [22] as follows: a) give visual access to the structural lesion, b) enable the surgeon to define and verify the extent of resection, and c) facilitate the protection of normal healthy tissue.

While early image guidance systems used frame-based stereotaxy, frameless stereotactic systems provide tools for accurate navigation by relating the location of instruments to preoperative, and more recently intraoperative image data. In the context of frameless stereotactic image guidance systems, computer vision techniques for image registration and segmentation are basic steps required for precise and interactive 3D rendering of the patient anatomy/physiology. An accurate stereotactic localization and digitization technique based on computer vision algorithms was reported in Heilbrun et al [23]. Their technique used a pair of 2D images acquired from different viewing angles with video cameras mounted on the ceiling of the operating room for determining the 3D location of markers seen in both images.

All stereotactic systems require co-registration, defined as a geometric mapping between at least two coordinate systems corresponding to the pre-operative and intra-operative spaces. A simple co-registration method requires known locations in both spaces of non-collinear points, defined either by fiducial markers or by natural landmarks. Computer vision algorithms for medical image registration also allow for rigid and non-rigid surface and contour matching. Pelizzari et al proposed in [24] a non-fiducial technique for registering MR, CT and PET brain images by matching the surface contour of the head. Anatomic surface curvatures may also be used for matching and registration purposes, as shown by Wang et al [25].

An interactive system for neurosurgery guidance and planning is presented in Gering et al [26]. Their system performs data fusion between various pre-operative and intraoperative scans by using 3D Slicer tools for image registration and segmentation. The system received positive clinical feedback for the graphical display of functional information, anatomical information, and information from contrast agents into a single view, as well as for the temporal data fusion. The main limitations were related to the surgeons' learning curve and to the use of the same interface for planning and for intra-operative guidance. The vast quantity of information extracted from data fusion and off-line analysis was found beneficial for planning, but overwhelming and distracting for on-line guidance.

In surgery guidance, image registration techniques play a central role; however, intraoperative image segmentation is likely to become a powerful tool in the process of image-guided interventions. Warfield et al [27] proposed a new intraoperative segmentation framework applied to the cryotherapy of liver cancer and to neurosurgery. Such a framework is designed to enable the monitoring of changes in anatomical structures (i.e. due to tumour resection) during surgery, and to quantitatively compare the progress of the interventional process with the preoperative plan.

3 Vision-Based Interfaces for the Operating Room

Computer Vision techniques such as markerless tracking of human motion and gesture recognition have been successfully integrated into perceptual user interfaces for applications such as video games, teleconferencing and surveillance. In particular, hand gesture recognition is useful for controlling the UI via command selection and virtual object manipulation. Surveys on hand gesture interpretation and on hand pose estimation are available in [28] and [29] respectively.

Operating rooms for minimally invasive surgery (MIS) are environments which could significantly benefit from using non-contact, gesture-controlled human-computer interfaces. Indeed, MIS procedures typically require computer support, and standard computer peripherals are difficult to sterilize. Therefore, standard clinical protocols involve a human assistant who manipulates the computer display according to the surgeon's commands and needs for visualization. However, the assistant-in-the-loop approach is suboptimal and sometimes leads to frustration and prolonged time for performing the intervention.

The non-contact mouse proposed by Graetzel et al [30] enables the surgeon to directly control the user interface with simple hand gestures. The non-contact mouse supports the "wait-and-click" and "push-to-click" paradigms by hand tracking and gesture classification. The gestures of interest are simple and based only on the 3D position of the right hand; the hand motion is mapped to pointer movement using non-linear gains, thus allowing for quick navigation and precise control. The non-contact mouse was successfully tested with a mock-up medical interface in the laboratory and in the operating room.

Face movement can also be used to control the user interface in MIS, as demonstrated by Nishikawa et al [31]. They proposed a face-tracking system that controls the laparoscopic camera positioning according to a face movement grammar. The tracker works with a 3DOF face pose by assuming that during the intervention, the surgeon's face remains almost parallel to and at a constant distance from the monitor screen. A user survey on a virtual testbed proved the usefulness of the system and its superiority over a voice-controlled interface; however, some tests suggested that face motion may distract the surgeon when performing very precise surgical actions.

While explicit user control over the graphical interface is absolutely necessary for the execution of critical tasks in laparoscopy, a certain degree of automation would speed up the intervention by allowing the surgeon to focus more on the surgery and less on the interface manipulation. As reported by Grange et al [32], process automation in a medical environment has to obey strict safety rules; thus, any automated user interface must be overridable by the surgeon's decision. The system proposed by Grange et al [32] combines gesture interpretation for explicit interaction with real-time monitoring of the surgeon's activity for automatically addressing some predictable surgeon's needs. They identify typical modes in a user interface for endoscopy, and prove that the transition from one mode to another can be automated using information from the visual tracking of the head, torso, and hands of the surgeon. The systems proposed in McKenna et al [33] and in Nishikawa et al [34] take a different automation approach by tracking instruments in the laparoscopic video instead of tracking the surgeon.

Video understanding techniques have also been developed for the quantitative assessment of basic surgical skills [35]. Such techniques may be good candidates for building multimodal HCIs which combine haptic [36] and visual information for evaluating laparoscopic and other surgical skills.

4 Perceptual Interfaces for Motor-Impaired Users

According to the Model Human Processor [37], a simple human computer interaction process comprises three cycles, namely perceptual, cognitive, and motor. Keates et al [38] proved that in standard UIs, motor impairment affects not only the motor cycle, but also introduces extra perception and cognitive cycles. To avoid extra cognitive loads, the design of HCI for motor-impaired users must consider two alternatives: adapting the content of the graphical display and/or customizing input systems for allowing a more natural interaction. Computer Vision techniques such as real-time tracking of body features are suitable for implementing the second alternative.

In Morrison and Mckenna [39], the hand motion trajectory represents a basis for learning and recognizing hand gestures. Their system uses HMM models to learn and recognize a user-defined set of simple hand gestures which replace basic commands in a standard web browser (back, forward, start, open, OK, refresh, cancel, and close).

The system proposed by Betke et al [40] is able to track the motion of diverse body parts (nouse, eyes, chin, foot) with an algorithm based on spatiotemporal template matching. The body feature to be tracked can be specified by the user in the initialization phase. The motion of the tracked feature is then mapped onto the motion of the mouse pointer on the screen. The system in [40] proved to be useful for interaction based on dynamic or static item selection by pointing without clicking.

A binary selection paradigm in a visual UI can be controlled with eye-blinks or eye-brow motion, as shown in Grauman et al [41]. Applications based on eye-blink/eyebrow-raises input do not require mouse movement, since they are entirely controlled by clicks. Therefore, such applications implement a scanning mechanism which displays one option at a time until the user selects the desired option with a long eye-blink or with an eyebrow-raise. The eye-blinks and eyebrow-raises are detected by algorithms based on template matching and on the properties of eye motion during blinking.

While web browsers and educational games are useful tools for communication and learning, another basic need of motor-impaired users is related to moving in their physical environment. Yanco and Gips [42] proved that electrode-based gaze-tracking can be successfully integrated in the design of intelligent wheelchairs. In Kuno et al [43], the user can control the motion of an autonomous wheelchair via a perceptual interface which detects changes in his face direction using computer vision algorithms. This interaction paradigm is more natural than using the conventional joystick, since humans usually look in the direction they want to take. Visual information is collected with two video cameras, one observing the user and the other observing the environment. This visual information is seamlessly integrated with information from other types of sensors specific to autonomous vehicles in order to achieve the right balance between autonomous and user-defined motion. The robotic wheelchair is also able to observe the user at a distance, and to respond to the user's commands by recognizing hand gestures. This option is more suitable for elderly persons with limited capability of walking. More information on the design of vision-based systems for the assistance of elderly persons is to be found in the next section.

5 Vision-Based Intelligent Systems for Elderly Assistance

Pervasive computing is a promising technology for supporting aging-in-place. Indeed, intelligent environments can assist elderly persons in a supportive and non-intrusive way during their daily activities. Moreover, automatic visual monitoring may detect abnormal harmful events such as falls, loss of balance, or suspect periods of inactivity possibly caused by a stroke. Sensing agents based on computer vision are unobtrusive, since they can be embedded in the environment without altering it. Vision algorithms are able not only to extract low-level data such as the subject's location and posture, but also to analyze human activities and interactions with the environment. Fig. 1 shows a generic vision-based monitoring system with its main modules, namely sensing, decision-making, and prompting.

However, the acceptance of vision-based monitoring in elderly health care is controversial, since it raises privacy and ethical concerns. User requirements for a fall detector in the context of a visual monitoring system were investigated by Mckenna et al [44]. They found that potential elderly users received well the idea of a vision-based monitoring system provided that images/videos are not stored or broadcast, and the visual input is analyzed only by a computer. Other findings in [44] were related to the design of the communication between the system and the faller (i.e. the prompting module in Fig. 1). Thus, the potential users wanted to be able to clear false alarms generated by the system, and also to press a button to call for help in case the fall has not been detected.

A significant segment of the community-dwelling elderly population suffers from various degrees of decline in cognitive functions and in memory. Such persons are unable to complete activities of daily living (ADL) [45] such as bathing and dressing on their own, since they do not remember the entire sequence of steps involved in the activity. A technique for visual ADL monitoring and assistance was proposed in Mihailidis et al [46]. This technique is applied to hand washing and integrates information from colour-based hand tracking and tracking of step-specific objects (i.e. soap bar). To date, computer vision techniques face limitations in fine motion tracking and ADL monitoring since they are very sensitive to contextual change. Pervasive computing systems such as the Aware Home project at Georgia Tech [47], the MIT's House_n project (http://architecture.mit.edu/house_n) and the Intel's Proact system [48] process information from various types of sensors for modelling ADLs. However, visual information plays an important role in activity recognition; indeed, [48] reported that hand washing is not well recognized by their system using radio-frequency identification tags, since water and metal absorb radio waves produced by these tags.

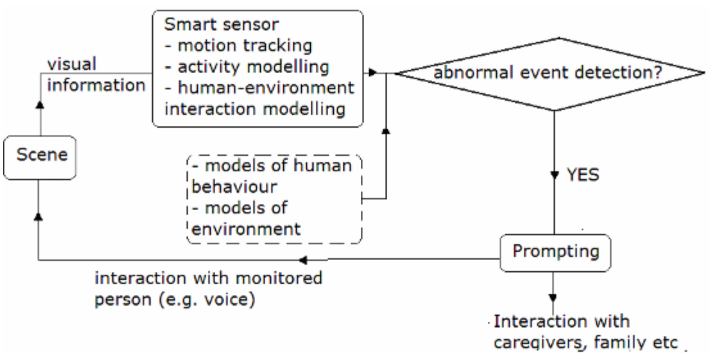

Smart sensor
- motion tracking
- activity modelling
- human-environment interaction modelling
visual information
abnormal event detection?
Scene
YES
- models of human behaviour
- models of environment
Prompting
interaction with monitored person (e.g. voice)
Interaction with caregivers, family etc

Fig. 1. Generic diagram of an intelligent system using a computer vision-based sensing agent

Visual monitoring of whole body motion finds relevant applications in systems designed to detect falls and unusual inactivity caused by stroke. The methods proposed by Nait-Charif and Mckenna [49] and by Sixsmith and Johnson [50] are based on a spatial model of the home environment composed of inactivity zones and entry (high traffic) zones. This model is learned in [49] from the spatio-temporal trajectories of the tracked subject and it can be used for fall and unusual activity detection. The approach in [50] is based on a user-defined spatial map, called risk map, and monitors subject with infrared cameras. Depending upon the zone type, their system adjusts the extent of acceptable zone inactivity. A subtle motion detector is correlated in [50] to the inactivity monitor in order to correctly detect some motion in static activities such as watching TV. Cuchiarra et al [51] proposed a video-based fall detector based on posture estimation and remotely connected with a PDA in order to enable an audio-video connection in case of emergency.

Whether focused on tracking fine motion for ADL assistance or on whole-body motion for fall or unusual inactivity detection, vision-based monitoring systems have to process data and prompt the users in real-time. A significant delay between the occurrence and the detection of an event is critical, since it propagates to the prompting module. Mihailidis et al [46] outlined that in ADL assistance, a delayed prompting for the completion of a particular step will result in user confusion. A late fall detection could have even more severe implications on the health and safety of the elderly.

6 Conclusions

In the context of user interface design, computer vision techniques play a dual role. First, computer vision algorithms for medical image understanding extract information from image data in order to provide assistance in processes such as diagnosis, therapy planning, and surgical navigation. *Such algorithms are designed to observe medical images.* As shown in Section 1, they may be successfully integrated in user interfaces for enhanced data visualization. Most important, differences in user requirements result in different algorithmic designs of computer vision techniques embedded in graphical interfaces, as well as in different interface design strategies. As an example, interactive and intuitive segmentation techniques in computer-aided diagnosis are preferred over automated techniques. Conversely, image-guided surgery uses 3D models of anatomical structures built from the off-line segmentation and multi-modal registration of the raw image data; thus, segmentation and registration are not controlled by the user, and may be automated. The emphasis in image-guided surgery is on the accuracy and reliability of the patient-specific models built from preoperative data.

A second category of HCI-related computer vision techniques are designed to observe the user and/or understand his actions. Most often, action recognition is based on tracking the user's body parts. Techniques belonging to this second category are useful for controlling perceptual interfaces such as discussed in Sections 2 and 3. Moreover, human motion analysis can play an important role as input data in intelligent systems for monitoring the well-being of seniors. In the design of perceptual interfaces, the ease-of-use and the real-time response are essential user requirements. The real-time response also plays a critical role in monitoring human activities for elderly assistance and support. As one may expect, other user requirements are task-oriented, such as addressing privacy and safety concerns in visual monitoring systems.

Computer vision techniques have already proven their usefulness for the design of medical research-oriented graphical user interfaces. It is expected that clinical acceptance will improve with advances of 'going filmless' in screening, diagnosis and therapy planning. The use of computers for inspecting images will probably trigger the use of interactive tools that augment visualization and improve the speed of diagnosis and planning.

Vision-based perceptual interfaces are currently among the latest trends in video games. As in the case of virtual reality applications, it is predictable that mature technologies and vision algorithms used in games will become transferable and/or adaptable to health applications such as described in Section 2 and 3. Multimodal interfaces integrating voice and visual recognition are also a promising alternative.

Vision-based monitoring for elderly assistance and support faces significant technological and ethical challenges. However, integrating video information with data gathered by other types of sensors, such as proposed in the design of intelligent environments can significantly improve the robustness of these systems. Respecting user requirements related to the privacy of video content, and promoting social connectedness via audio/video communications are viable strategies in coping with ethical concerns.

References

[1] J. Gosbee, E. Ritchie, "Human-computer interaction and medical software development." Interactions 4(4): 13-18, 1997.

[2] D. Pham, C. Xu, J. Prince, "Current methods in medical image segmentation." Annual Review of Biomedical Engineering, 2: 315-337, 2000.

[3] S.D. Olabarriaga , A.W.M. Smeulders, "Interaction in the segmentation of medical images: A survey." Medical Image Analysis 5: 127–142, 2001.

[4] G. Gerig, M. Jomier, M. Chakos, "Valmet: A new validation tool for assessing and improving 3D object segmentation." Proc. MICCAI Conf. Med. Image Comput. and Computer-Assisted Intervention, 516-523, 2001.

[5] S. Warfield, K.H. Zou, W. M. Wells, "Simultaneous Truth and Performance Level Estimation (STAPLE): An Algorithm for the Validation of Image Segmentation." IEEE Trans. on Med. Imag. 23 (7): 903-921, 2004.

[6] L. O'Donnell, C.-F. Westin, W.E.L. Grimson, et al, "Phase-based user-steered image segmentation", Proc. MICCAI Conf. Medical Image Computing and Computer-Assisted Intervention, 1022–1030, 2001.

[7] W. A. Barrett, E. N. Mortensen, "Interactive live-wire boundary extraction." Med. Image Anal., 1(4):331–341, 1997.

[8] P. J. Elliott, J. Diedrichsen, K. J. Goodson, R. Riste-Smith, G. J. Sivewright, "An object-oriented system for 3D medical image analysis." IBM Systems Journal, 35 (1): 4 – 24, 1996.

[9] P.J.Elliot, J.M. Knapman, W. Schlegel, "Interacting segmentation for radiation treatment planning." IBM Systems Journal, 31 (4): 620–634, 1992.

[10] M. Meissner, K. Zuiderveld (organizers), G. Harris, J. R. Lesser, A. Persson, M. Vannier (panelists), "End Users' Perspectives on Volume Rendering in Medical Imaging: A job well done or not over yet?" Panel, IEEE Visualization, Vis 05, Minneapolis, USA Oct. 2005.

[11] R. Shahidi, L. Clarke, R.D. Bucholz, H. Fuchs, R. Kikinis, R.A. Robb, M. Vannier, “White paper: challenges and opportunities in computer-assisted intervention.” Comp. Aided Surgery, 6(3): 176-181, 2001.

[12] M. Meissner, B. Lorensen, K. Zuiderveld, V. Simha, R. Wegenkittl, “Volume Rendering in Medical Applications: We’ve got pretty images, what’s left to do?” Panel, IEEE Visualization, Vis 02, Boston, USA Oct. 27-Nov. 1, 2002.

[13] D. T. Gering, A. Nabavi, R. Kikinis, et al, “An Integrated Visualization System for Surgical Planning and Guidance Using Image Fusion and Interventional Imaging.” Proc. MICCAI Conf. Medical Image Computing and Computer-Assisted Intervention, 809–819, 1999.

[14] R. Robb, *Biomedical Imaging, Visualization and Analysis*, John Wiley and Sons, Inc: New York, 1999.

[15] L. Ibanez, W. Schroeder, L. Ng, J. Cates, *The ITK Software Guide*, 2003.

[16] D. Hanson, R. Robb, et al, "New software toolkits for comprehensive visualization and analysis of 3D multimodal biomedical images." Journal of Digital Imaging. 10(2), 1-2, 1997.

[17] K. Augustine, D. Holmes, R. Robb, “ ITK and Analyze: A synergistic integration”. Proc. SPIE Medical Imaging. (2004) pp. 6–15.

[18] J. Rexilius , W. Spindler, J. Jomier, et al,“ A Framework for Algorithm Evaluation and Clinical Application Prototyping using ITK.” MICCAI Workshop on Open-Source Software 2005.

[19] N. Hanssen, B. von Rymon-Lipinski, T. Jansen, et al “Integrating the Insight Toolkit *itk* into a Medical Software Framework.” Proc. of CARS Computer Assisted Radiology and Surgery 2002.

[20] C. Alberola-Lopez, R. Cardenes, M. Martin et al, “diSNei: A collaborative environment for medical images analysis and visualization”. Proc. of MICCAI Medical image computing and computer-assisted interventions. pp. 814–23, 2000.

[21] F. Simmross-Wattenberg, N. Carranza-Herrezuelo, C. Palacios-Camarero et al, “Groupslicer: a collaborative extension of the 3D-slicer.” Journal of Biomed. Informatics, 38: 431-442, 2005.

[22] J. McInerney, D.W. Roberts, “Frameless Stereotaxy of the Brain”. The Mount Sinai Journal of Medicine, 67(4):300-310, 2000.

[23] M.P. Heilbrun, P. McDonald, C.Wicker et al, “Stereotactic localization and guidance using a machine vision technique”. Stereotactic Functional Neurosurgery, 58:94-98, 1992.

[24] C.A. Pelizzari, G.T.Y. Chen, D.R. Spelbring et al, “Accurate three-dimensional registration of CT, PET, and MR images of the brain”, Journal Comput. Assist. Tomography, 13:20-26, 1989.

[25] Y. Wang, B.S. Peterson, L.H. Staib, 3D Brain Surface matching based on geodesics and local geometry, Computer Vision Image Understanding, 89:252-271, 2003.

[26] D.T. Gering, A. Nabavi, R. Kikinis et al, “An Integrated Visualization System for Surgical Planning and Guidance Using Image Fusion and an Open MR”. Journal of Magnetic Resonance Imaging, 13:967–975, 2001.

[27] S. Warfield, A. Nabavi, T. Butz et al, “Intraoperative segmentation and non-rigid registration for image-guided therapy”. Proc. of MICCAI, Medical Image Computing and Computer-Assisted Intervention, 176-185, Oct. 2000.

[28] V.I. Pavlovic, R. Sharma, T.S. Huang. "Visual interpretation of hand gestures for human computer interaction: A review," IEEE Trans. on Patt. Anal. and Machine Intelligence, 19 (7), 1997.

[29] A. Erol, G. Bebis, M. Nicolescu, R. Boyle, X. Twombly, "A Review on Vision-Based Full DOF Hand Motion Estimation", Proc. of the IEEE Workshop on Vision for Human-Computer Interaction (V4HCI), San Diego, Ca, June 2005.

[30] C. Graetzel, T.Fong, S.Grange, C.Baur, "A non-contact mouse for surgeon-computer interaction". Technology and Health Care, 12(3), 2004.

[31] A. Nishikawa, T. Hosoi, K. Koara et al, "Face mouse: a novel human-machine interface for controlling the position of a laparoscope", IEEE Trans. on Robotics and Automation, 19(5):825-844, 2003.

[32] S. Grange, T. Fong, C. Baur, "M/ORIS: A medical/operating room interaction system". Proc. of the ACM Int. Conf. on Multimodal Interfaces, State College, PA, 2004.

[33] S.J. McKenna, H. Nait Charif, T. Frank, "Towards Video Understanding of Laparoscopic Surgery: Instrument Tracking", Proc. of Image and Vision Computing, New Zealand, 2005.

[34] A. Nishikawa, S. Asano, R. Fujita et al "Robust visual tracking of multiple surgical instruments for laparoscopic surgery," Proc. of Comp. Assisted Radiology and Surgery, London, 2003.

[35] J. Chen, M. Yeasin, R.Sharma, "Visual modeling and evaluation of surgical skill". Pattern Analysis and Applications, 6:1-11, 2003.

[36] J. Rosen, M. Solazzo, B. Hannaford, M. Sinanan, "Objective Evaluation of Laparoscopic Skills Based on Haptic Information and Tool/Tissue Interactions". Computer Aided Surgery, 7(1): 49-61, 2002.

[37] S.K. Card, T.P. Moran, A. Newell, *The Psychology of Human-Computer Interaction*, 1983 Hillsdale, NJ: Lawrence Erlbaum Associates.

[38] S. Keates, P.J. Clarkson, P. Robinson, "Developing a methodology for the design of accessible interfaces." Proc. of the 4th Workshop on User Interfaces for All, Stockholm, Sweden, 1998.

[39] H. Morrison and S. J. McKenna, "Contact-free recognition of user-defined gestures as a means of computer access for the physically disabled". Proc. 1st Workshop on Univ. Access and Assistive Technology, Cambridge, UK, 99–103, Mar. 2002.

[40] M. Betke, J. Gips, P. Flemming, "The Camera Mouse: Visual tracking of body features to provide computer access for people with severe disabilities", IEEE Trans. on neural systems and rehabilitation eng., 10(1): 1-9, 2002.

[41] K. Grauman, M. Betke, J. Lombardi, J. Gips, G.R. Bradski, "Communication via eye-blinks and eye-brow raises: video-based human-computer interfaces", Univ. Access. Inf. Soc., 2: 359-373, 2003.

[42] H.A. Yanco, J. Gips, "Preliminary investigation of a semi-autonomous robotic wheelchair directed through electrodes," Proc. Rehab. Eng. Soc. of North Am.Annual Conf., 414-416, 1997.

[43] Y. Kuno, N. Shimada, Y. Shirai, "Look where you're going: A robotic wheelchair based on the integration of human and environmental observations." IEEE Robotics and Automation Magazine, 27-34, March 2003.

[44] S. J. McKenna, F. Marquis-Faulkes, P. Gregor, A. F. Newell, "Scenario-based drama as a tool for investigating user requirements with application to home monitoring for elderly people," Proc. of HCI Int., Crete, Greece, June 2003.

[45] S. Katz, "Assessing Self-Maintenance: Activities of Daily Living, Mobility, and Instrumental Activities of Daily Living." J. Am. Geriatrics Soc., 31(12): 721–726, 1983.

[46] A. Mihailidis, B. Carmichael, J. Boger, "The use of computer vision to support aging-in-place, safety, and independence in the home." IEEE Trans. on Inf. Tech. in Biomed., 8(3):238-247, 2004.

[47] Abowd, G. A. Bobick, I. Essa, E. Mynatt, W. Rogers, "The aware home: developing technologies for successful aging.", AAAI technical report.

[48] M. Philipose, K. Fishkin, M. Perkowitz et al, "Inferring activities from interactions with objects." IEEE Pervasive Computing, 3(4):50-57, 2004.

[49] H. Nait-Charif, S. J. Mckenna, "Activity summarization and fall detection in a supportive home environment", IEEE Int. Conf. on Pattern Recognition, 323-326, 2004.

[50] A. Sixsmith, N. Johnson, "A smart sensor to detect the falls in the elderly", IEEE Pervasive Computing, 3(2):42-47, 2004.

[51] R. Cucchiara, C. Grana, A. Prati, R. Vezzani, "Computer vision techniques for PDA accessibility of in-house video surveillance", Proc. of ACM Int. Workshop on Visual Surveillance IWVS, Berkeley (CA), 87-97, 2003.

Multiple Hypothesis Target Tracking Using Merge and Split of Graph's Nodes

Yunqian Ma[1], Qian Yu[2], and Isaac Cohen[1]

[1] Honeywell Labs, 3660 Technology Drive, Minneapolis, MN 55418
{yunqian.ma, isaac.cohen}@honeywell.com
[2] Institue for Robotics and Intelligent Systems, University of Southern California
qianyu@usc.edu

Abstract. In this paper, we propose a maximum a posteriori formulation to the multiple target tracking problem. We adopt a graph representation for storing the detected regions as well as their association over time. The multiple target tracking problem is formulated as a multiple paths search in the graph. Due to the noisy foreground segmentation, an object may be represented by several foreground regions and one foreground region may corresponds to multiple objects. We introduce merge, split and mean shift operations that add new hypothesis to the measurement graph in order to be able to aggregate, split detected blobs or re-acquire objects that have not been detected during stop-and-go-motion. To make full use of the visual observations, we consider both motion and appearance likelihood. Experiments have been conducted on both indoor and outdoor data sets, and a comparison has been carried to assess the contribution of the new tracker.

1 Introduction and Background

Multiple target tracking is a key component in visual surveillance. Tracking provides a spatio-temporal description of detected moving regions in the scene, this low level information is critical for recognition of human actions in video surveillance. In the considered visual tracking problem, the observations used are the detected moving blobs. Incomplete observations due to occlusions, stop and go motion or noisy foreground detections constitute the main limitation of blob-based tracking methods. We propose a tracking method that allows to split, merge detected moving regions, as well as re-acquiring moving targets after a stop-and-go motion.

Several problems need to be addressed by a tracking algorithm: A single moving object (*e.g.* one person) can be detected as multiple moving blobs. In this case the tracking algorithm needs to 'Merge' the detected blobs. Similarly, one detected blob can be composed of multiple moving objects, in this case the tracking algorithm needs to 'Split' and segment the detected blob into corresponding moving objects. The split and merge of detected blobs has to be robust to partial or total occlusions, as well as being capable of differentiating detected moving regions of nearby objects. Stop-and-go motion, or non-detection due to similarity of the object to the background may require the tracker to re-acquire the target.

Moveover, the detected blobs could be due to erroneous motion detection. Here the tracking algorithm needs to filter these observations in presence of static or dynamic occlusions of the moving objects in the scene. Finally the number of moving objects in the scene vary as new moving objects enter or leave the field of view of the camera.

A large number of tracking algorithms have been developed in the past decades. Several data association tracking algorithms have been proposed ranging from a simple nearest neighbor association to the complex multiple hypothesis tracker [8][7]. The Probabilistic data association (PDA) method [13], which is considered a good compromise between performance and complexity, uses a weighted average of all the measurements within the tracks' validation gate [14] to estimate the target state. The PDA method deals with multiple targets as independent objects in term of observations, and therefore less suitable for addressing the situations where multiple observations correspond to a single target and vice versa. JPDAF [6] is an extension of the PDA, where the measurement of target association probabilities is evaluated jointly across the targets. The Multiple Hypothesis Tracker (MHT) tracking algorithm was first developed by Reid [7] and propagated multiple hypotheses in time. The ranking of the hypotheses requires evaluating over all existing hypotheses and thus pruning and merging were used to reduce the set of hypothesis to a manageable size.

Most of existing data association algorithms cannot address the merge or split of the observations for an accurate estimation of the target state. The multiple hypothesis trackers are the most widely used, however these methods assume a one-to-one mapping between observations and targets. An attempt to extend these frameworks to merge and split behaviors was proposed in [15], which introduced the concept of virtual measurement to represent the splitting and merging of detected regions. However, the association was inferred using a brute force method. [16] performs a multi-object segmentation using a probabilistic pixel classification algorithm which uses the appearance model to calculate the likelihood of a pixel to belong to a particular object. An iterative approach then finds the front-most model first and deletes it from the foreground object and then fits the second object. [17] defines an occlusion relation parameter for addressing the blob splitting problem.

In this paper, we formulate the multiple target tracking problem as a maximum a posteriori (MAP) problem. We expand the set of observations with hypothesis added by merge, split and mean shift operations, which are designed to deal with noisy foreground segmentation due to occlusion, foreground fragment and missing detection. All these added hypotheses will be validated during the MAP estimation. The remainder of this paper is organized as follows: In Section 2, we present the MAP formulation for multiple target tracking problem and the considered motion and appearance likelihood models. In Section 3 we describe our proposed multiple hypothesis method with merge, split and mean shift operations. In Section 4, we present experimental results obtained on real video surveillance sequences and discuss the proposed method. In Section 5 we conclude the paper.

2 Multiple Target Tracking Formulation

In a multiple target tracking problem, the objective is to track multiple target trajectories over time given a set of noisy measurements provided by a motion detection algorithm. The observations considered are blobs that cannot be regarded as punctual observations, and furthermore, the targets position and velocities are automatically initialized and do not require operator interaction. The detector usually provides image blobs which contain both the estimated location, size and the appearance information as well. Within any arbitrary time span $[0, T]$, there are K unknown number of targets in the monitored scene. Let $y_t = \{y_t^i : i = 1, ..., n_t\}$ denote the observations at time t, $Y = \cup_{t \in \{1,...,T\}} y_t$ is the set of all the observations within the duration $[0, T]$. The multiple target tracking can be formulated as finding the set of K best paths $\{\tau_1, \tau_2, \cdots, \tau_K\}$ in the temporal and spatial space, where K, represents the number of moving objects or targets in the scene and this number is unknown. We denote a track by the set of its observations: $\tau_k = \{\tau_k(t) : t \in [1, T]\}$ where $\tau_k(t) \in y_t$ represents the observation of track τ_k at time t.

We utilize a graph representation $G =< V, E >$ of all measurements within time $[0, T]$. The graph is a directed graph that consists of a set of nodes $V = \{y_t^k : t = 1, \cdots, T, k = 1, \cdots, K\}$. We also consider one special measurement of y_t^0 to represent the null measurement at time t which corresponds to missed detections. A directed edge $(y_{t_1}^i, y_{t_2}^j) \in E, t_1 < t_2$ is defined between two nodes based on proximity and similarity of the corresponding detected blobs. The weight or cost associated to an edge will be computed using to motion and appearance models described in the following paragraphs. To reduce the amount of edges defined in the graph we consider only edges for which the weight or cost (motion and appearance) between two nodes is more than a pre-determined threshold. An example of such a graph is shown in Fig. 1. At each time instant, there are m_t observations. The one which doesn't belong to any track represents a false alarm. The shaded node represents a missing observation, inferred by the tracking.

We formulate the multiple target tracking problem as a MAP problem, given the observations over time find K best paths $\tau^*_{1,\cdots,K}$ through the graph G, as follows:

$$\tau^*_{1,\cdots,K} = \text{argmax}(P(\tau_{1,\cdots,K}|Y)) \tag{1}$$

The posterior of the K best paths can be represented as the observation likelihood of the K paths and the prior of the K paths as in [2]. A prior distribution model of $P(\tau_k : k = 1, \cdots, K)$ widely used in data association algorithms [5,6], is represented as follows:

$$P(\tau_{1,\cdots,K}) = \prod_{i=1}^{T} p_d^{T_m^i} (1 - p_d)^{K - T_m^i} p(F_m^i) \tag{2}$$

where T_m^i is the number of measurements associated to the tracks and F_m^i is the number of measurements not associated to the tracks. $p(F_m^i)$ is a Poisson distribution of F_m^i, and p_d denotes the detection rate which can be estimated

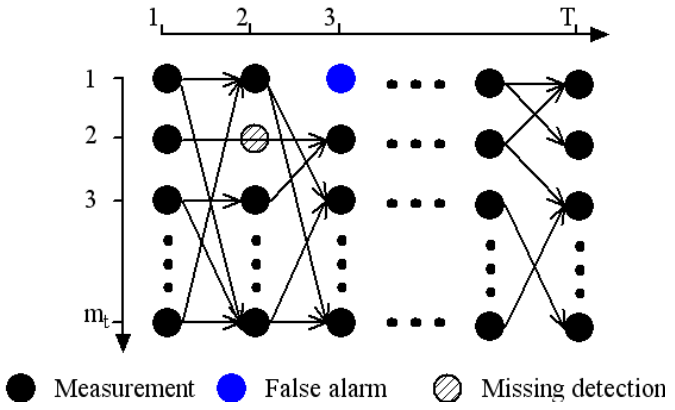


Fig. 1. Graph representation of measurements

from prior knowledge of the detection procedure. By introducing this prior, the posterior of the unknown K paths can be represented as:

$$P(\tau_{1,\cdots K}|Y) \propto P(Y|\tau_{1,\cdots K})P(\tau_{1,\cdots K}) \tag{3}$$

The K paths multiple target tracking can be extended to a MAP estimate as follows:

$$\tau^*_{1,\cdots K} = \arg\max(P(Y|\tau_{1,\cdots K})P(\tau_{1,\cdots K})) \tag{4}$$

Since our measurements are image blobs, beside position and dimension (width and height) information, an appearance model is considered in the proposed traking method. To make full use of available visual cues, we consider both motion and appearance likelihood measures. By assuming that each target is moving independently, the joint likelihood of the K paths over time $[1, T]$ can be represented as:

$$P(Y|\tau_{1,\cdots K}) = \prod_{k=1}^{K} P_{\text{motion}}(\tau_k(1), \cdots, \tau_k(T))P_{\text{color}}(\tau_k(1), \cdots, \tau_k(T)) \tag{5}$$

The joint probability is defined by the product of the appearance and motion probabilities. This probability maximization approach is inferred using the Viterbi algorithm [10].

Motion Model. We consider a constant velocity motion model in the 2D image plane and 3D ground plane. We denote x^k_t the state vector of the target k at time t to be $\left[l_x, l_y, w, h, \dot{l}_x, \dot{l}_y, l_{gx}, l_{gy}, \dot{l}_{gx}, \dot{l}_{gy}\right]$ (location, width, height and velocity in 2D image, and location on the ground plane). We consider a linear kinematic model:

$$x^k_{t+1} = A^k x^k_t + w^k \tag{6}$$

where x_t^k is the state vector for the target k at time t, A^k is the transition matrix and we assume w^k to be normal probability distributions, $w^k \sim N(0, Q^k)$. Here we use a constant velocity motion model. The observation $y_t^k = [u_x, u_y, w, h, u_{gx}, u_{gy}]$ contains the measurement of a target position and size in 2D image plane and position on 3D ground plane. Since observations often contain false alarms, the observation model is represented as:

$$y_t^k = \begin{cases} H^k x_t^k + v^k & \text{if it belongs to a target} \\ \delta_t & \text{false alarm} \end{cases} \tag{7}$$

where y_t^k represents the measurement which may arise either from a false alarm or from the target, and δ_t is the false alarm rate at time t. We assume v^k to be normal probability distributions, $v^k \sim N(0, R^k)$.

The measurement is modelled as a linear model of current state if it belongs to a target otherwise it is modelled as a false alarm δ_t, which is assumed to be a uniform distribution.

Let $\hat{\tau}_k(t_i)$ denote the *a posteriori* state estimate and $\hat{P}_t(\tau_k)$ the *a posteriori* estimate of the error covariance matrix of τ_k at time t. Along a track τ_k the motion likelihood of one edge $(\tau_k(t_1), \tau_k(t_2)) \in E, t_1 < t_2$ can be represented as $P_{\text{motion}}(\tau_k(t_2)|\hat{\tau}_k(t_1))$. Given the transition and observation model in Kalman filter, the motion likelihood then can be written as:

$$P_{\text{motion}}(\tau_k(t_2)|\hat{\tau}_k(t_1)) = \frac{1}{(2\pi)^{3/2}\ \det(\hat{P}_{t_2}(\tau_k))} \exp\left(\frac{-e^T \hat{P}_{t_2}^{-1}(\tau_k) e}{2}\right) \tag{8}$$

where $e = y_t^k - HA^{t_2 - t_1}\hat{\tau}_k(t_1)$ and$\hat{P}_{t_2}(\tau_k)$can be computed recursively by a Kalman filter as $\hat{P}_{t_2}(\tau_k) = H(A\hat{P}_{t_2-1}(\tau_k)A^T + Q)H^T + R$.

Appearance model. The tracking of each region relies on the kinematic model described above, as well as on an appearance model. The appearance of each detected region is modelled using a non-parametric histogram All RGB bins are concatenated to form a one dimension histogram. The appearance likelihood between two image blobs $(\tau_k(t_1), \tau_k(t_2)) \in E, t_1 < t_2$ in track k, is measured using the a symmetric Kullback-Leibler (KL) divergence defined as follows.

$$P_{\text{color}}(\tau_k(t_2)|\hat{\tau}_k(t_1)) = \frac{1}{2} \sum_{c=r,g,b} (P_i(c) - P_j(c)) \log(\frac{P_i(c)}{P_j(c)}) \tag{9}$$

Other appearance models such as [9][11] can be used by this framework as well.

Given the motion and appearance models, we associate a weight to each edge defined between two nodes of the graph. This weight combines the appearance and motion likelihood models presented in the previous paragraphs.

In Eq.7 and Eq.9, we assume the state of target at time t is determined by the previous state at time $t-1$ and the observation at time t is a function of the state at time t alone, *i.e.* Markov condition. Thus the joint likelihood of K paths in Eq.5 can be factorized as follows:

$$P(Y|\tau_{1,\cdots K}) = \prod_{k=1}^{K} \prod_{(t_1,t_2)\in\tau_k}^{T} P_{\text{motion}}(\tau_k(t_1)|\hat{\tau}_k(t_2))P_{\text{color}}(\tau_k(t_1)|\hat{\tau}_k(t_2)) \quad (10)$$

where $(\tau_k(t_1), \tau_k(t_2))$ represents the edge in the track k.

3 Augmented Graph Representation for Multiple Hypothesis Tracker

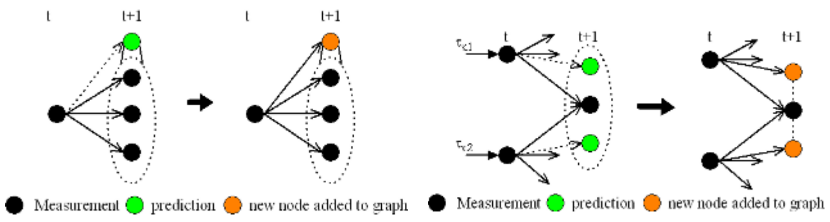
Most multiple target tracking algorithms [1][8][4] assume that no two paths pass through the same observation. This assumption is reasonable when considering punctual observations. However, this assumption is usually violated in the context of visual tracking problem, where the targets cannot be regarded as points and the inputs to the tracking algorithm are usually image blobs. In the following paragraphs we present an extension of the PDA framework to handle split and merge behaviors in estimating the best paths.

3.1 Merge and Split Hypothesis

The proposed merge and split behaviors correspond to a recursive association of new observation, given estimated trajectories. At given time instant t, we have obtained K best paths which are denoted as $[\tau_1^t, \cdots, \tau_K^t]$. Using this estimated tracks, we can evaluate how the m_{t+1} observations $\{y_{t+1}^i : i = 1, ..., m_{t+1}\}$ at time $t+1$ fit the estimated tracks which end at time t. The spatial overlap between estimated state at instant time t and new observation will considered as a primary cue and we consider the following two cases:

– If the prediction of $\bar{\tau}_k^t(t+1)$ has a sufficient spatial overlap with more than one observation at time $t+1$. This will trigger a *merge* operation which merges the observations at time $t+1$ into one new observation. This new observation carrying the merge hypothesis will be added into the graph. The merge operation is illustrated in Fig.2(a).
– If the predicted positions and shapes of more than one track spatially overlap with one observation y_{t+1}^* at time $t+1$. The set of candidate tracks is κ,$|\kappa| > 1$. This will trigger a *split* operation, which splits the node y_{t+1}^* into several observations. These observations, which encode the split hypothesis, will be added to the observations set at time $t+1$. The split operation proceeds as follows: for each track τ_k^t in κ whose prediction has a sufficient overlap with y_{t+1}^*:
  - Change the predicted size and location at time $t+1$ to find the best appearance score $s_k = P_{\text{color}}(\bar{\tau}_k^t(t+1), y_{t+1}^*)$;
  - Create a new observation node for the track with the largest s_k and add it to the graph;
  - Reduce the confidence of the area occupied by the newly added node and recompute the score s_k for each track left in κ;

  Iterate this process until all candidate tracks in κ that overlapped with the observation y_{t+1}^* are tested. The split operation is illustrated in Fig.2(b).



(a) Hypothesis added by merge operation

(b) Hypothesis added by split operation

Fig. 2. Merge and split hypothesis added to the graph

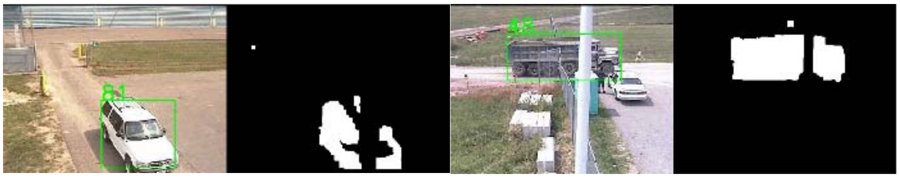
3.2 Mean Shift Hypothesis

Noisy segmentation of the foreground regions often provides incomplete observations not suitable for a good estimation of the position of the tracked objects. Indeed, moving objects are often fragmented, several objects are merged into a single blob, and regions are not detected in the case of stop-and-go motion.

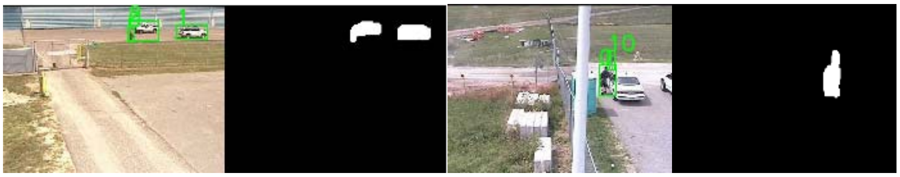
We propose to incorporate additional information from the images for improving appearance-based tracking. Since at each time t, we have already maintained the appearance histogram of each target, we introduce the mean shift operation to keep track of this appearance distribution when the motion blob does not provide good enough input. Mean shift method [12], which can be regarded as a mode-seeking process, was successfully applied to the tag-to-track problem. Usually the central module of mean shift tracker is based on the mean shift iterations to find the most probable target position in the current frame according to the previous target appearance histogram. In our multiple target tracking problem, if a reliable track is not associated with a good observation at time t, due to a fragmented detection, non detection or large mismatch in size, we instantiate a mean shift algorithm to propose the most probable target position given the appearance histogram of the track. Note that the histogram used by mean shift is established using past observations along the path (within a sliding window), instead of using only the latest one. Using the predicted position from mean shift, we add a new observation to the graph. The final decision will be made by considering all the observations in the graph. To prevent the mean shift tracking from tracking a target after it leaves the field of view the mean shift hypothesis is considered only for trajectories where the ratio of real node to the total number of observations along the track is larger than a threshold.

4 Experiment Results

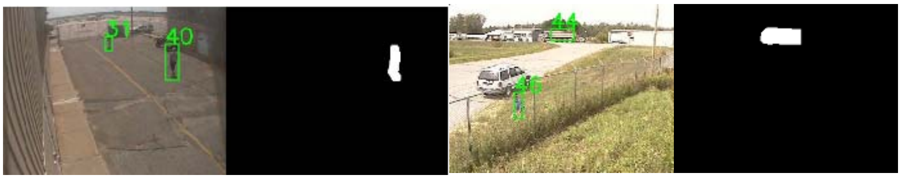
In the considered experiments we used a sliding temporal window of 45 frames to implement our algorithm as an online algorithm. The graph contains the observations between time t and $t + 45$. When new observations are added into graph, the observations are older than t will be removed from the graph.



(a) Tracking result with merge operation when foreground regions fragment



(b) Tracking result with split operation when foreground regions merge



(c) Tracking result with mean shift operation when missing detection happens

Fig. 3. Experiment results of Merge and split hypothesis added to the graph

We tested our tracking algorithm on both indoor and outdoor data sets. The data considered were collected inside our lab, around the parking lots and other facilities at Honeywell. In the considered data set a large number of partial or complete occlusions between targets (pedestrians and vehicles) were observed. In the experimental tests conducted, the input considered for the tracking algorithm were the foreground regions and the original image sequence. We have tested the accuracy of the proposed tracking algorithm and have compared it to classical PDA without the added merge, split and mean shift hypothesis.

Fig.3 shows the data sets with tracking results overlaid and the foreground detected. Due to the noisy foreground segmentation, the input foreground for one target could have multiple fragment regions, shown in Fig3.(a). In the case where two or more moving objects are very close to each other, we may have a single moving blob for all the moving objects, shown in Fig.3(b). The case that targets merge into background is shown in Fig.3(c). Given the homography between the ground plane and the image plane, the targets can be tracked on

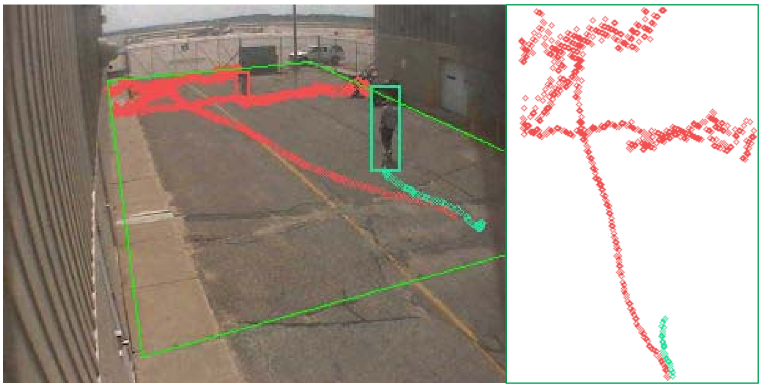


Fig. 4. Tracking targets using ground plane information. Left, estimated trajectories are plotted in the 2D image. Right, the positions of the moving people in the scene are plotted on the ground plane.

the 3D ground plane, shown in Fig.4. Without any code optimization, the time performance of our online tracking algorithm with 45-frame sliding window is close to real time (15-20 fps for 3-5 targets) on a P4 3.0Hz.

5 Conclusion

In this paper, we presented a method for multiple targets tracking in video surveillance. If we partition the application scenarios into easy, medium and difficult cases, most of the existing tracking algorithms can handle the easy cases relatively well. However for the medium and difficult cases, multiple targets could be merged into one blob especially during the partial occlusion and one target could be split into several blobs due to noisy background subtraction. Also missed detections happen often in presence of stop and go motion, or when we are unable to distinguish foreground from background regions without adjusting the detection parameters to each sequence considered.

We have introduced a mechanism based on of multiple hypothesis which expands the solution space. The proposed formulation of multiple target tracking problem as a maximum posterior(MAP) and the expanded set of hypothesis by considering merge, split and mean shift operations is more robust. It deals with noisy foreground segmentation due to occlusion, foreground fragments and missing detections. Experimental results show good performance on tested data sets.

References

1. B. LaScala and G. W. Pulford, *Viterbi data association tracking for over-the-horizon radar*, Proc. Int. Radar Sym., Vol.3, pp. 155-164, Sept. 1998.
2. K. Buckley, A. Vaddiraju, and R. Perry,*A new pruning/merging algorithm for MHT multitarget tracking*, Radar–2000 , May 2000

3. T. Quach and M. Farooq, *Maximum likelihood track formation with the Viterbi Algorithm* Proc. 33-rd IEEE Conf. on Decision and Control, pp. 271-276, Dec. 1994.
4. D. Castanon, *Efficient algorithms for finding the K best paths through a trellis* IEEE Trans. on Aerospace& Elect. Sys., Vol. 26, No. 2, pp. 405-410, Mar. 1990.
5. Fortman, T.E., Bar-Shalom, Y., and Scheffe, M. *Sonar tracking of Multiple Targets Using Joint Probabilistic Data Association*, IEEE Journal of Oceanic Engineering, Vol. OE-8, No.3, July 1983, pp. 173-184
6. Y. Bar-Shalom and T.E. Fortmann. *Tracking and Data Association* Mathematics in Science and Engineering Series 179 Academic Press, San Diego, CA, 1988.
7. D.B. Reid. *An algorithm for tracking multiple targets* In IEEE Transaction on Automatic Control, volume 24(6),pp. 843854, December 1979.
8. I.J. Cox and S.L. Hingorani *An Efficient Implementation of Reid's Multiple Hypothesis Tracking Algorithm and Its Evaluation for the Purpose of Visual Tracking*, PAMI,
9. J. Kang, I. Cohen, and G. Medioni. *Object reacquisition using invariant appearance model.* In ICPR, pp. 759762, 2004. 18-2,1996,138-150
10. J. Kang, I. Cohen and G. Medioni.*Continuous Tracking Within and across Camera Streams*, IEEE, Conference on CVPR'2003, Madison, Wisconsin. 2003
11. J. Kang, I. Cohen and G. Medioni. *Persistent Objects Tracking Across Multiple Non Overlapping Cameras, IEEE Workshop on Motion and Video Computing, MOTION'05. Breckenridge, Colorado. Jan 4-5,2005.*
12. C.Yizong, *Mean Shift, Mode Seeking, and Clustering,* PAMI, vol. 17, no. 8, pp. 790-799, Aug., 1995.
13. S. Blackman,*Multiple Target Tracking with Radar Applications* Artech House, 1986.
14. Y. Bar-Shalom, and E. Tse, *Tracking in a Cluttered Environment with Probabilistic Data Association*, Automatica, , pp. 451-460, 1975.
15. A. Genovesio and J. Olivo-Marin. *Split and merge data association filter for dense multi-target tracking*, In ICPR,pages IV: 677680, 2004.
16. A. Senior ,*Tracking People with Probabilistic Appearance Models*, PETS02, 2002, pp.48-55
17. Wu. Ying, Y. Ting and H. Gang, *Tracking Appearances with Occlusions*, in Proc. IEEE Conf. on CVPR'03, Vol.I, pp.789-795, Madison, WI, June, 2003

Understanding 3D Emotions Through Compact Anthropometric Autoregressive Models

Charlotte Ghys[1,2], Nikos Paragios[1], and Bénédicte Bascle[2]

[1] MAS - Ecole Centrale Paris, Grande Voie des Vignes, 92295 Chatenay-Malabry, France
[2] Orange - France Telecom R&D, 2 avenue Pierre Marzin, 22300 Lannion, France

Abstract. Reproducing realistic facial expressions is an important challenge in human computer interaction. In this paper we propose a novel method of modeling and recovering the transitions between different expressions through the use of an autoregressive process. In order to account for computational complexity, we adopt a compact face representation inspired from MPEG-4 standards while in terms of expressions a well known Facial Action Unit System (FACS) comprising the six dominant ones is considered. Then, transitions between expressions are modeled through a time series according to a linear model. Explicit constraints driven from face anthropometry and points interaction are inherited in this model and minimize the risk of producing non-realistic configurations. Towards optimal animation performance, a particular hardware architecture is used to provide the 3D depth information of the corresponding facial elements during the learning stage and the Random Sampling Consensus algorithm for the robust estimation of the model parameters. Promising experimental results demonstrate the potential of such an approach.

1 Introduction

Facial expressions are a critical aspect of inter-human communication and provide a good basis for understanding emotional conditions. Such understanding can be associated with certain geometric deformations of a neutral expression that can vary significantly across subjects. Realistic reproduction of emotions is a critical aspect of human-computer-interaction with numerous applications like games, e-learning, etc.

The aim of this paper is produce an anthropometric mathematical framework capable of explaining the deformations of one expression to another through a combination between prediction and image-driven stereo inference. Such problem has been active in the area of computer vision for almost three decades now. In 1972, the first facial model was introduced [14], a purely 2D approach that was based on a polygonal approximation of the face on which emotion frames/representations have been extracted and used as key-frames of animations through interpolation. Such a method requires storing different face expressions as key-frame and assumes that direct interpolation between such expressions can produce realistic animations. More advanced methods to facial modeling assume dense facial models like deformable meshes [11,5]. Such models aim to introduce the notion of muscle contraction simulation through the use of spline-based parameterization [15], free form deformations [13], etc... Other methods have considered the use of biomechanical facial models. Such models are then coupled with stereo data determined either through reconstruction from multiple views [7] or using laser/range scanners. An alternative refers to the use of 3D Morphable Models [2,12], combined with the well known 2D Active Appearance Model [4] introduce 2D+3D Active Appearance Models [16]. However such methods are expensive in terms of complexity.

Once models have been learnt, understanding facial expressions becomes a parameter estimation problem between the obtained geometry and the model expressing a neutral facial condition. These methods suffer from being computationally expensive because of the complexity of the model as well as due to the difficulty of obtaining dense stereo information from multiple images. In order to address such limitations, the idea of face modeling through a number of key 3D points was considered. The well-known MPEG-4 [1,9] introduces a parameter coding for 3D face synthesis and animation. It permits to animate a downloaded face model using a very low bandwidth. However such a model still suffers from explicit interpolation between states often leading to unrealistic geometric and photometric animations.

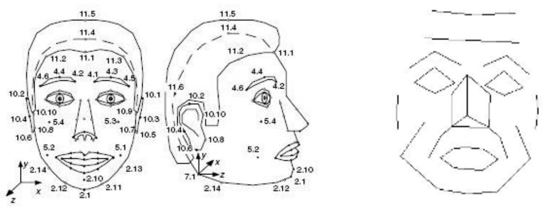
In this paper we propose a novel approach for facial expression reproduction as well as facial animations. Such a method reformulates expression in the form of time series where transition between different expressions is modeled through an autoregressive matrix. In order to overcome limitations related with dense stereo estimation, we consider a variant of the MPEG-4 facial model. Such a model is coupled with a prediction mechanism where anthropometric constraints are explicitly introduced to account for facial geometric properties (symmetry, etc.) and points interaction exploited to improve the emotion reality. Markers are used to obtain a training set where reconstruction and depth estimation for the key points is feasible. Last, but not least, Random Sampling Consensus and robust regression is used to account for the presence of outliers as well as errors in the reconstruction.

The reminder of this paper is organized in the following fashion. In section 2 we detail the facial model and we present the acquisition process. The autoregressive animation framework is presented in section 3. Experimental results and validation are part of the later section.

2 Anthropometric Model and Facial Expressions

Understanding facial expressions consists of estimating a number of parameters that explain the current state of the face model. Such an action requires on one hand the definition of a neutral state for the face, and on the other hand the range of parameters explaining the different expressions. The selection of the model is critical in this process in terms of performance quality and computational cost. Since the aim of our paper is the understanding of facial expressions, one can conclude that a compromise between complexity and performance is to be made aiming a model that is able to capture the most critical deformations.

We adopt the MPEG-4 standard model to represent faces [1]. MPEG aimed in an audio and video coding representation where its MPEG-4 version is extended to multimedia including images, text, graphics, 3-D scenes, etc ... and particularly face and body animation. Such a model consists of 205 degrees of freedom as shown in [Fig. (1.i)] and is able to capture expressions. It consists of introducing the neutral state through a set of



(i) (ii)

Fig. 1. (i) MPEG-4 face model using a number of control points positioned with aim to capture and reproduce facial animations, (ii) simplified MPEG model towards understanding facial expressions

Features Points (FPs). The model is still quite complex while the estimation of the actual positions of the control points through image inference is being quite problematic. We further simplify such a model using a smaller number of control points positioned in critical aspects of the face [Fig. (1.ii)]. Such a simplification is guided from representation of expressions using geometric deformations as well as hardware acquisition constraints, leading to a model that consists of 114 degrees of freedom and refers to a simple 3D polygonal approximation of the face.

2.1 Modeling Facial Expressions

Paul Ekman and Wallace F. Friesen, designed in 70's a muscle-based [6] Facial Action Units System (FACS) to help human observers in depicting facial expression using a verbal way. Such a system includes 44 Actions Units (AU) (inner brow raiser, lower lip depressor, etc ...), expressing all possible facial movements (comparable to phonemes in speech). Each of them is related to the contraction of one or several muscles and each facial expression can be described as an activation of one or a set of AUs.

An alternative to muscle-based understanding of facial expressions through AU is a description according to geometrical changes of the face. Such an approach is more tractable and reformulates expression understanding in more technical fashion. To this end, several geometric models have proposed with the MPEG-4 standard and its facial animation parameters (FAPs) ([1] and [9]) being the most popular [Fig. (2)]. Such an animation mechanism consists of set of (FPs) associated with a set of FAPs, corresponding to a particular facial action deforming a face model. A MPEG-4 client is so able to animate FPs of a downloaded face model using compressed FAPs. MPEG-4 standard provides some hints for FPs location on a neutral face, as equality (FP 4.3, the uppermost point of the left eyebrow, is defined as the midpoint between FPs 4.1 and 4.5, left eyebrow corners, in the upper eyebrow contour, such as $4.3x = \frac{4.1x+4.5x}{2}$) or inequality ($11.3x > 4.3x$ supposed that 11.3 point x-coordinates is greater than 4.3 point x-coordinates). These constrains are specified in [Tab. (1)].

One of the most important limitations of such a model is the lack of anthropometric constraints. Indeed the most obvious physiognomy constraints is the face symmetry. Even if a face is not exactly symmetric, considering the 6 basic emotions defined in Fig. 2, the movements of FPs are rather symmetric with certain errors that are not critical.

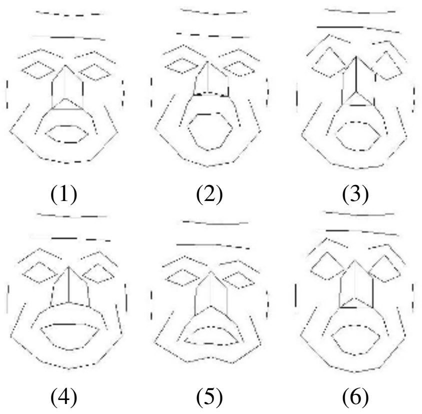


Fig. 2. (1) Anger, (2) Disgust, (3) Fear, (4) joy, (5) Sadness, (6) Surprise. Those six emotions are expressed according to Ekman and Friesen's work.

We assume that when expressing joy, if the left corner of the mouth goes up and the right corner goes up too, with the same intensity. The correspondence of symmetric FPs are defined in the last column of [Tab. 1]. Furthermore, one can see that motion vertical direction is often far more important than the one observed in the horizontal one. Therefore additional constraints are introduced in the vertical direction aiming to minimize the risk of erroneous estimation. To this end, as it seems that some points are correlated to each other, or in other words, have similar movements, it is judicious to exploit this observation to add those new constraints to our model. The displacement of a point is so computed relatively to the more correlated points. This novel approach, with more constraints, gives more robustness to our model.

Once such a model has been considered, the next step consists of the acquisition of training data that could be used in a learning stage towards reproducing transitions between expressions. Our acquisition process can be decomposed in three stages: preparation, capture, reconstruction.

2.2 Acquisition System for Learning Data Set

Once the face model (or rather Feature Points and their position) is defined, the FPs are projected using a projector on the face, while the head is placed in an adjustable tubes frame (See Fig. 3). This permits to place the markers, exactly at the same place at each capture session with the same person.

Then, to obtain a 3D data set, a classical stereo system is used. The cameras are "one above the other" (one camera gives a straight-on viewpoint and the other a low angle viewpoint). It permits to keep the face symmetry and avoids hidden parts, particularly around the nose. Video are captured at the frame rate of 50 fps.

The reconstruction stage begins by landmarks tracking using meanshift tracking algorithm based on color similarity [3]. After FPs positions and correspondences are known in both images, classic 3D reconstruction from images [7] is used to compute FPs 3D positions, i.e. knowing points position m and m' in both images , point M 3D

Table 1. Recommended location constraints for few FPs on a face model 1 in term of coordinates and new symmetric constraints

#	Description	Constraints recommended by MPEG-4	Symmetric point
4.1	Right corner of left eyebrow		4.2
4.2	Left corner of right eyebrow		4.1
4.3	Uppermost point of the left eyebrow	$x = (4.1x + 4.5x)/2$	4.4
4.4	Uppermost point of the right eyebrow	$x = (4.2x + 4.6x)/2$	4.3
4.5	Left corner of left eyebrow		4.6
4.6	Right corner of right eyebrow		4.5
11.1	Middle border between hair and forehead		
11.2	Right border between hair and forehead	$x < 4.4x$	11.3
11.3	Left border between hair and forehead	$x > 4.3x$	11.2

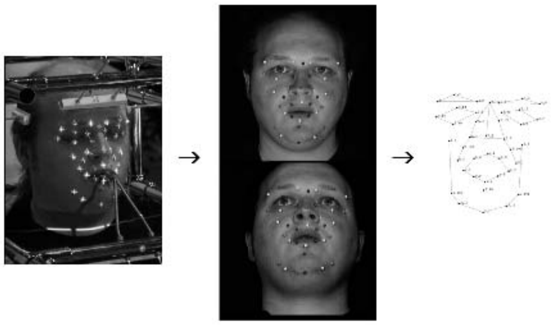


Fig. 3. Training data set acquisition set-up using a projector and markers

coordinates can be recover resolving the equations : $m = PM$ ans $m' = P'M$, where P and P' are matrices projection of both cameras. An overview of the system is given in Fig. 3.

3 Inference Between Observations and Auto Regressive Animation Models

Building predictive models is equivalent with expressing the state of a random variable $\mathbf{X}(t)$ in time as a function of the previous system using a linear or a non-linear model:

$$\mathbf{X}(t) = G(\mathbf{X}(t-k); k \in [1, p]) + \eta(t) \tag{1}$$

with p known to be the order of the model and η a noise model that is used to describe errors in the estimation process. Such a process can be decomposed in a learning stage and a prediction step. In the first step, given a set of sequences of observations and the selection of the prediction mechanism we aim to recover the parameters of this function such that for the training set, the observations meet the prediction.

The Auto Regressive Process (ARP) permits to solve the problem of predicting objects position in time and is able to model the temporal deformation of a high-dimensional vector. In the context of this paper, such a vector corresponds to the position of the face control points. For this first approach, we assume that a linear system can express the transitions between expressions, where a finite ARP consists of:

$$\mathbf{X}(t) = \mathbf{H} \cdot \begin{bmatrix} \mathbf{X}_{t-1} \\ \vdots \\ \mathbf{X}_{t-p} \end{bmatrix} + \eta(t) \tag{2}$$

where $\mathbf{H}$ is often called the prediction matrix. In the case of transitions between facial expressions, we consider a simple linear model that assumes a linear behavior in time leading to the following expression,

$$\mathbf{X}(t) = \sum_{k=1}^{p} H^k \mathbf{X}_{t-k} + \eta(t) \tag{3}$$

In order to obtain invariance to pose/depths changes, we first determine a rigid transformation (global translation, rotation and scale) between the current and the previous states of the model to a reference 3D configuration, the neutral face. Since explicit correspondences between the control points of the face are attainable the estimation of such a transformation can be done in a straightforward fashion. In order to account for outliers (important motion due to certain facial expressions), a robust error metric (M-estimator [10]) is used along with the Euclidean distance between 3D points:

$$\inf_{T_i} \sum_{k=1}^{n} \rho(|\overline{\mathbf{X}}^k - \mathbf{X}_{i,T}^k|) \tag{4}$$

with k being the degrees of freedom of the model and ρ a robust estimator. The calculus of variations of such a cost function will lead to a linear system guiding the estimation of the transformation T. Once such parameters have been defined the next task consists of determining the actual pose-invariance prediction matrix. It is natural to assume that displacements in the different axes are relatively independent. Furthermore, given that the number of frames needed to transit from one expression to another is relatively small (4-6), we consider a low order AR model (order 1). These assumptions lead to the following simplified AR model

$$\begin{aligned} \mathbf{X}(t) &= H^x \cdot \mathbf{X}_{t-1} + \eta_x(t) \\ \mathbf{Y}(t) &= H^y \cdot \mathbf{Y}_{t-1} + \eta_y(t) \\ \mathbf{Z}(t) &= H^z \cdot \mathbf{Z}_{t-1} + \eta_z(t) \end{aligned} \tag{5}$$

Let suppose now N FPs, considering each FP separately, it is possible to compute the position as a linear combination of the same point at previous time. This means that

H^x, H^y and H^z are $N \times N$ diagonal matrices. Such an assumption does not encode certain facial geometric constraints like symmetry, therefore certain modifications on the form of H^x, H^y and H^z could be considered to introduce such conditions.

Towards introducing explicit constraints driven from the facial geometry in vertical direction, we separate the face points in two regions : the upper face and the lower face. One can assume that points in each region are correlated. Mouth points movements influence jaw movements and eyebrow movements influence forehead movements. In both regions and for each emotion, correlation coefficients between points displacements are computed. If the correlation coefficient between two points i and j is greater than a threshold τ, the points are assumed to be correlated, and they are mutually dependent during the displacement calculation process. This correlation can be modeled within the autoregressive model as follows :

$$H^y(i,j) = \begin{cases} 1 + \frac{U(\mu_{i,j})d_{i,j}}{\sum_{j=0}^{N} U(\mu_{i,j})} & \text{if } i = j \\ \frac{U(\mu_{i,j})d_{i,j}}{\sum_{j=0}^{N} U(\mu_{i,j})} & \text{otherwise} \end{cases} \quad \text{where} \quad U(\mu_{i,j}) = \begin{cases} \mu_{i,j} & \text{if } \mu_{i,j} > \tau \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

and $H^y(i,j)$ is the matrix H^y element at the i^{th} line and j^{th} column, $\mu_{i,j}$ is the correlation coefficient between point i and point j, $d_{i,j}$ is the point i displacement between time t and time $t+1$ according to point j position at time t.

Then, prediction is projected in symmetric and MPEG-4 constraints space, introduced in section 2.1 and specified in [Tab. (1)]. One can assume that when two points are symmetric, they have the same displacement in y and z-direction while they move in opposite directions in the x-axis. So, considering two points i and j, their movements are modelized by the i^{th} line H_i and the j^{th} line H_j in modelization matrices. Supposing i and j are symmetric, the line H_i and the line H_j become :

$$\hat{H}_i^x = \frac{H_i^x - H_j^x}{2} \quad \text{and} \quad \hat{H}_j^x = \frac{-H_i^x + H_j^x}{2} \tag{7}$$

while the i^{th} and j^{line} in H^y and H^z are replaced by their mean. The same way to proceed is used with MPEG-4 constraints. Considering three points i, j and k, such as k is in the middle between i and j on horizontal axis, the k^{th} line H_k^x become the mean of the lines H_i^x and H_j^x. Those equations are used in parameter estimation step to add constraints to the model.

3.1 Parameter Estimation

There are several methods to perform inference for the parameters H^k,such as $k = 1, \ldots, p$ as the Euler-Lagrange equations, the Yule-Walker estimation, the Levinson-Durbin algorithm, etc ... In the context of our approach one can observe that the estimation process is over-constrained. Furthermore, due to the simplistic nature of the model capturing the most important parameters of the animation process cannot be done through global least-square estimators.

The RANdom SAmple Consensus (RANSAC) algorithm is chosen to estimate ARP parameters. The RANSAC algorithm is based on the assumption that data set is made of ”inliers”, which can explain the model, and ”outliers” which don’t fit the model. It is

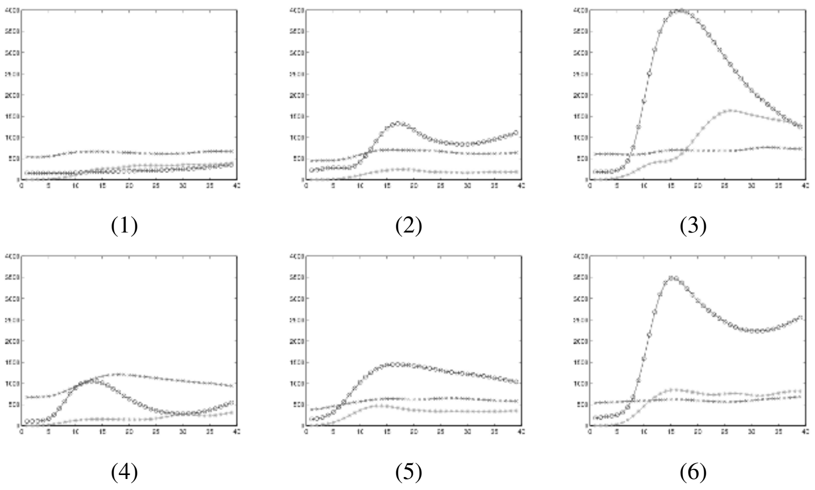


Fig. 4. Error during the sequence in L^2 norm. ($\times$) : x-coordinates, ($\circ$) : y-coordinates, ($*$) : z-coordinates, in black mean of the 3 coordinates. (1) Anger, (2) Disgust, (3) Fear, (4) joy, (5) Sadness, (6) Surprise.

Table 2. Mean error in L^2 norm for a point in pixel in function of face zone and emotion

	Anger	Disgust	Fear	Joy	Sadness	Surprise
Front	7.1	7.4	11.8	7.4	9	10.6
Eye brows	7	7.2	13.7	6.1	7.7	12.3
Eyes	3.9	5.8	9.1	4.6	4.5	8
Nose	3.9	4.2	8.6	5	3.7	6.6
Smiling wrinkles	3.5	4.5	5	7.8	3.6	5.9
Mouth	3.4	5.1	4.9	4.9	6.3	6.7
Jaw	3.5	4.5	5.7	5.7	7	5.6
Ears	1.9	3.6	3.1	3.4	4.5	4.9

particularly suitable in this case considering people don't express facial emotion with the same intensity and speed, and one can't really define when an emotion begins and when it ends. The RANSAC algorithm was proposed by Fischler and Bolles in [8]. Model parameters are estimated using a random subset of the dataset, supposed to be inliers. These parameters are then validated using the the rest of the data set, and the configuration leading to the lowest inference error between the model and the entire training set is retained.

Our adapted version consist of computing the ARP parameters H^x, H^y and H^z according to the randomly defined subset of the data set (3D points coordinates during the emotion expression process) to obtain first $N \times N$ matrices, taking into account correlation constraints. Then, the prediction is projected in the symmetric and equality constraints space as described in [Tab. (1)]. Towards improving model performance, the acceptance criterion on one hand consists of the approximation error while at the same time such a solution should satisfy the inequality constraints as defined in [Tab. (1)].

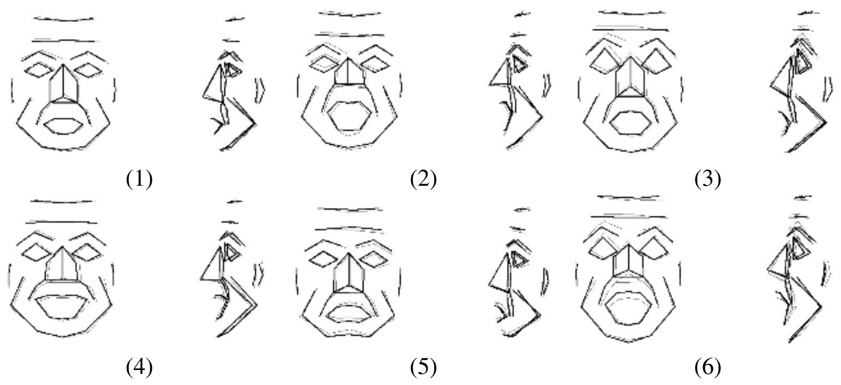


Fig. 5. Last frame of the sequence. In gray thin line, real model, in black thick line, estimated model. (1) Anger, (2) Disgust, (3) Fear, (4) joy, (5) Sadness, (6) Surprise.

4 Conclusion and Discussion

In this paper we have proposed a new method to model and reproduce realistic facial emotions. Our method is based on a compact face representation driven from the MPEG standards, and aims on reproducing facial animations through anthropometric prediction models. To this end, explicit geometric constraints on the facial form are used to determine the form of the prediction function. Robust estimation to account for global depth changes, as well as for the presence of outliers through the use of RANSAC are introduced to determine the parameters of such a function. Promising experimental results using a small number of control points to represent the face as well as simple predictive mechanisms demonstrate the potentials of our approach.

In order to validate the proposed method, 6 expressions were used, using four different subjects. On the first three subjects, parameter learning by inference was performed, while on the last subject the predictive mechanism was used to reproduce the intermediate states of animation. Euclidean distances between the control points among expressions were used to determine a quantitative validation measure as shown in [Tab. (2)] while the face configuration as determined from the model next to actual one is presented in [Fig. (5)] for a qualitative evaluation of the obtained results. Furthermore, the evolution of the prediction error for all coordinates spaces is shown in [Fig. (4)]. While one can claim that the errors seem acceptable, we can also observe that the method mostly fails on the upper face area. Furthermore, due to the linearity of the model as well as the relatively low order, the method seems to underperfom when important non uniform motion is observed along frames.

Future work consists of automating the acquisition process. In the current state markers is used to determine correspondences. Learning 2D as well as 3D patches corresponding to the geometry and appearance of these points could automate the process. Furthermore, non-linear models of transitions are to be investigated towards more efficient capturing of transitions. Last, but not least the use of HMMs towards a more complete probabilistic animation schema could improve the performance of the method in a qualitative fashion.

Acknowledgements. The authors would like to thank Gaspard Breton from Orange-FT R&D, Rennes for providing facial stereoscopic data with markers.

References

1. *ISO/IEC IS 14496-2 Visual*. 1999.
2. V. Blanz and T. Vetter. A Morphable Model for the Synthesis of 3d Faces. In *SIGGRAPH*, page 187194, 1999.
3. D. Comaniciu, V. Ramesh, and P. Meer. Kernel-Based Object Tracking. In *Pattern Analysis Machine Intelligence*, volume 25, pages 564–575, 2003.
4. T. Cootes, G. Edwards, and C. Taylor. Active Appearance Models. In *Pattern Analysis and Machine Intelligence*, 2001.
5. D. DeCarlo and D. Metaxas. Optical Flow Constraints on Deformable Models with Applications to Face Tracking. In *International Journal of Computer Vision*, July 2000.
6. P. Ekman and W.V. Friesen. *Facial Action Coding System*. Palo Alto, 1978.
7. O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, 1993.
8. M.A. Fischler and R.C. Bolles. Random Sample Concensus : a Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. In *Communications of the ACM*, volume 24, pages 381–395, 1981.
9. R. Forchheimer, I.S. Pandzic, and et al. *MPEG-4 Facial Animation: the Standards, Implementations and Applications*. John Wiley & Sons, 2002.
10. P. Huber. Robust Estimation of Location Parameter. In *Annals of Mathematical Statistics*, 1964.
11. S. Ilic and P. Fua. Generic Deformable Implicit Mesh Models for Automated Reconstruction. In *ICCV workshop on Higher-Level Knowledge in 3D Modelling and Motion Analysis, Nice, France*, October 2003.
12. Z. Liu, Z. , Zhang, C. Jacobs, and M. Cohen. Rapid Modeling of Animated Faces from Video. In *International Conference on Visual Computing*, pages 58–67, september 2005.
13. N. Magnenat-Thalmann, E. Primeau, and D. Thalmann. Abstract Muscle Action Procedures for Human Face Animation. *The Visual Computer*, 3, 1987.
14. F.I. Parke. Computer Generated Animation of Faces. In *ACM National Conference*, pages 451–457, 1972.
15. C.L.Y. Wang and D.R. Forsey. Langwidere : A New Facial Animation System. In *Computer Animation*, pages 59–68, 1994.
16. J. Xiao, S. Baker, I. Matthews, and T. Kanade. Real-time Combined 2d+3d Active Appearance Models. In *Conference on Computer Vision and Pattern Recognition*, 2004.

Graph-Based Multi-resolution Temporal-Based Face Reconstruction

Charlotte Ghys[1,2], Nikos Paragios[1], and Bénédicte Bascle[2]

[1] MAS - Ecole Centrale Paris, Grande Voie des Vignes, 92295 Chatenay-Malabry, France
[2] Orange - France Telecom R&D, 2 avenue Pierre Marzin, 22300 Lannion, France

Abstract. Reproducing high quality facial expressions is an important challenge in human-computer interaction. Laser-scanners offer an expensive solution to such a problem with image based alternatives being a low-resolution alternative. In this paper, we propose a new method for stereo reconstruction from multiple video pairs that is capable of producing high resolution facial models. To this end, a combinatorial optimization approach is considered and is coupled in time to produce high resolution depth maps. Such optimization is addressed with the use of graph-cuts leading to precise reconstruction of facial expressions that can then be used for animation.

1 Introduction

Facial expressions play a fundamental role in inter-human communication and are a keen element of mixed reality systems with a number of applications like human computer interaction. Shape (3D) reconstruction [7] from images have been a well studied problem, in particular for short base-line binocular camera systems. Based on constraints driven from the epipolar geometry, a number of methods were proposed to recover 3D structure. Simple correlation based techniques, dynamic programming [15], space carving [11], variational and level set methods [8] as well as combinatorial methods like graph-cuts [4,10,3] are some of the state-of-the-art methods in the literature for object reconstruction. Several other techniques are also specialized in face reconstruction refering to the use of 3D Morphable Models [2,12], deformable meshes [9,6] or muscle contraction simulation through the use of spline-based parameterization [16], free form deformations [13], etc... Other methods have considered the use of biomechanical facial models.

However photo-realistic faces are difficult to compute with low cost material as the definition of webcam images arent sufficient. The objective of this paper is to present a Super Resolution Reconstruction based on super resolution image reconstruction and graph cuts where successive frames are linked using optical flow and stereo reconstruction satisfies a number of subsequent stereo pairs. This paper is organized as follows. In Section 2, basic stereo view geometry is presented. Section 3 gives briefly basic notion of stereo, matching by graph cut. The method for accurate face reconstruction from low resolution images is described in section 4. Finally, our experimental results are shown in section 5 and discussed in section 6.

2 Basic Stereo View Geometry

In this section, we introduce the relation between the two cameras, then we explain rectification, briefly stereo matching and 3D reconstruction. In the next section, we come back and explain an advanced technique of stereo matching.

Basic notions from 3D geometry explain such a process. Given a 3D point M, its projection in a stereo system are m and m' such as shown in Fig.(1). m' is on the projection of m line of sight $l'_m = Fm$. l'_m is called epipolar line and F is the fundamental matrix. It encodes the relationship between the two images and all the corresponding points should satisfy: $m'^T Fm = 0$.

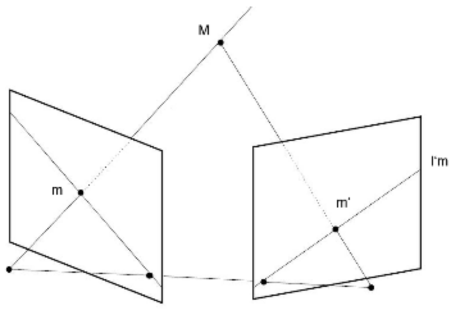


Fig. 1. Epipolar geometry

Calibration process is about to infer positions of points in one image from positions in the world This is modeled by the projection matrix P such as $m = PM$. P can be decomposed as follows : $P = A[Rt]$, where A describes the characteristics of the camera (focal length, location of the image center, real pixel size and distortion of the lens), and $[Rt]$ is a concatenation of a rotation matrix and a translation vector describing the change of world coordinate system.

Once fundamental matrix F is known, it can be used to constrain the correspondence search in one dimension. To simplify and speed up the stereo matching the image are warped so the epipolar lines become scanlines, a process that is known as rectification. Two corresponding points m and m' become :

$$m_r = \begin{pmatrix} x \\ y \end{pmatrix}, \quad m'_r = \begin{pmatrix} x + d \\ y \end{pmatrix} \tag{1}$$

where d it the horizontal displacement called disparity. Once epipolar lines have been determined, the stereo problem is simplified to horizontal correspondences. Since we want to reconstruct photo-realistic faces, we need a high resolution reconstruction. So we introduce an implicit super resolution stereo matching. Details are given in the sfollowing sections. Once we have the disparity for each pixel and we know the intrinsic and extrinsic parameters of the camera, we can compute the 3D position of the points by solving the system :

$$m = A[Rt]M = PM, \quad m' = A'[R't']M = P'M. \tag{2}$$

3 Stereo Reconstruction and Graph Cut

In this section, 3D Reconstruction [7] from images and graph cut method is presented .

Given a 3D point M, its projection in a stereo system are m and m'. Given $m = (x, y)$, matching problem is to determine $m' = (x + dx, y + dy)$. The relationship between both cameras is encoded by the fundamental matrix F such as all the corresponding points should satisfy: $m'^T Fm = 0$. Knowing F it is possible to know on which line on the second image, m' lies. This line is called an epipolar line. It can be used to constrain the correspondence search in one dimension. To simplify and speed up the stereo matching the image are warped so the epipolar lines become scan lines, a process that is known as rectification. Two corresponding points m and m' have now the same y-coordinates, and $x_{m'} = x_m + d$ where d is the horizontal displacement called disparity. Once epipolar lines have been determined, the stereo problem is simplified to horizontal correspondences.

Numerous methods exist in the literature to recover such correspondences. We consider a global optimization approach where internal smoothness constraints on the correspondences space are introduced. To this end, we adopt a discrete representation of the depth map, and a MRF-based stereo formulation. Global optimization techniques like Graph-cuts are then used to derive the optimal solutions.

3.1 Graph Cut in General

Let us now briefly introduce some notions from combinatorial optimization, namely the graph cut approach. Let G be a graph, consisting of a set of nodes V and a set of directed edges E that connect them such as $G = (V, E)$. The nodes set V contains two special terminal nodes which are called the source, s, and the sink, t. All edges in the graph are assigned some nonnegative weight or cost. A cut C is a partitioning of the nodes in the graph into two disjoint subsets S and T such that the source s is in S and the sink t is in T. The cost of a cut $C = (S, T)$ is defined as the sum of the costs of boundary edges (p, q) where $p \in S$ and $q \in T$. The minimum cut problem on a graph is to find a cut that has the lowest cost among all possible cuts. One of the fundamental results in combinatorial optimization is that the minimum cut problem can be solved by finding a maximum flow from the source s to the sink t.

Under certain conditions, one can prove that any optimization problem of the following form:

$$E = E_{data} + E_{smooth} \tag{3}$$

can be converted to a min cut/max flow problem. E represents the energy to minimize, and corresponds here the cost of the cut C. The definition of the data and smoothness terms depend on the problem to be solved. In the case of stereo reconstruction, the matching between intensities after applying the selected disparity component can be used as a data fidelity term. On the other hand smoothness often reflect the assumption that neighborhood pixels in the image correspond to the same depth level.

3.2 Graph Cut for Stereo-matching Problem

In a stereo-matching problem, the graph is forming a 3D-mesh see Fig.(2) and a cut should be view as a hyper surface. In this case, a vertex on the graph corresponds, to a possible match between two images. Each of these vertices is six-connected. Two connections, the ones in depth, correspond to other possible disparity values (connections of the same pixel in image 1 with neighbors along the epipolar line in image 2). The cost of these edges are $c_{p,d} = |I_1(x,y) - I_2(x+d,y)|$, the difference of intensity between pixel p with coordinates(x,y) in image 1, and pixel $(x+d,y)$ in image 2. It represents the data term of the energy we want to minimize. $E_{data} = \sum c_{p,d}$ where one end of $c_{p,d}$ is in S and the other one is in T.

The four other edges are connections with the four neighbors in the image introducing a smoothness connectivity. As a scene is most of time considered as piecewise smooth, their cost is usually defined for simplicity by a Potts model $v_{p,q} = u_{p,d} \cdot T(d_p \neq d_q)$ where d_p and d_q are disparities of pixels p and q, and

$$u_{p,d} = \begin{cases} 2K & \text{if } |I_p - I_q| \leq U \\ K & \text{if } |I_p - I_q| > U \end{cases} \tag{4}$$

Such a potential function tolerates certain discrepasies between depth levels within local neighborhood while penalizing heavily more important deviations. And so $E_{smooth} = \sum v_{p,q}$ where q is in S and p is in T. However this $v_{p,q}$ definition cant't be used for face reconstruction, as a face can not be considered like a piecewise constant scene. Obviously because it is an object, but also because it is made of curves and there are just few crests. Furthermore a difference of pixel intensity doesn't mean there is a difference of depth (an example of this is the skin space between eyebrows. It is very light compared to eyebrows, but the depth is the same). A good alternative to E_{smooth} term is given in the following section.

4 Optical Flow Estimation

Optical flow estimation is equivalent with recovering a pixel-wise deformation field $(u(x,y), v(x,y))$ that creates visual correspondences between two consecutive images f and g. One can consider the Sum of Squared Differences (SSD) as the data-driven term to recover the deformation field (u,v) at the pixel level;

$$E_{data}(u,v) = \iint_{\Omega} (f(\mathbf{x}) - g((x,y) + (u(x,y), v(x,y))))^2 dxdy \tag{5}$$

Such an error norm is very sensitive to occlusions as well as to outliers and therefore it can be replaced with a robust estimator, or like an an M-estimator. Such a method assigns weights to the constraints at the pixel level that are disproportional to their residual error therefore rejecting the motion outliers. to this end, one should define the influence function, $\psi(x)$ like for example the Tukey's estimator :

$$\rho(x) = \begin{cases} x(K_\sigma - x) & \text{if } |x| < K_\sigma \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where K_σ characterizes the shape of the robust function and is updated at each iteration leading to the following cost function:

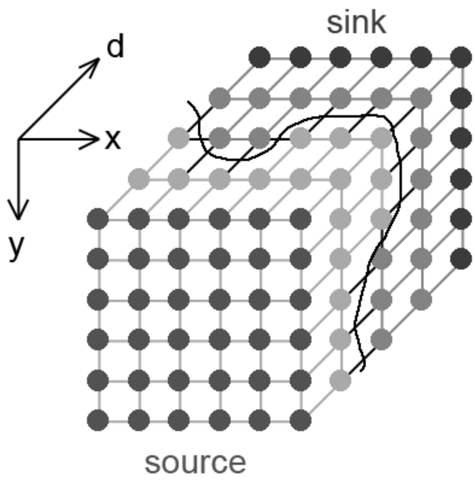


Fig. 2. Example of a graph for a stereo-matching problem

$$\begin{aligned} E_{data}(u,v) &= \iint_{\Omega} \rho(r)\, dxdy \\ &= \iint_{\Omega} \rho(f(\mathbf{x}) - g((x,y) + (u(x,y), v(x,y))))\, dxdy \end{aligned} \tag{7}$$

While such a model can be quite efficient it still suffers from the aperture problem. One can consider additional constraints to the constant brightness assumption like the gradient preservation assumption, recently introduced in [5]. Such a constraint that improves the estimation of the optical flow on the object boundaries where the visual constancy assumption is often violated.

5 Super Resolution Face Reconstruction

5.1 Redefinition of Local Consistency Towards Exploiting Facial Geometry

As mentioned in Section 3 the Potts model, used to define E_{smooth}, is not well designed for face reconstruction and must be redefined. To enforce E_{data} term, $v_{p,q}$is derived from the matching cost in depth of the two vertices that it links.

$$v_{p,q} = k(c_{p,d} + c_{p,d+1} + c_{q,d} + c_{q,d+1}) \tag{8}$$

It depends on the cost to assign the same disparity for both pixels. A larger k increases the smoothness of the reconstruction and avoids the outliers.

Such a global approach can lead to more consistent 3D face models. Its performance depends on the resolution of input images. Model accuracy is important in a number of applications where input images suffer from low resolution. The use of multiple stereo pairs taken from the same camera set-up in time can be used to provide more accurate reconstructions. The idea behind such an approach is that due to the discretization process each stereo pair will be able to capture different parts of the 3D surface. Putting such temporal reconstructions together under the assumption of correspondences between images can lead to more precise depth maps.

5.2 Super Resolution Image Reconstruction

In computer vision application, like medical, surveillance or satellite imaging, high resolution images are often required. Such images correspond to important pixel density where image is more detailed. On can find in [14] a good overview of different super-resolution techniques. The basic idea of all methods is to use multiple low resolution (LR) images captured from the same scene. These LR images are different representations of the scene, and considering there is no significant change between them, they are shifted from each other with a sub pixel precision.(a pixel unit precision would involved the information is the same in all images). Therefore, as soon as the scene motion can be estimated, super resolution (SR) image reconstruction is possible. Furthermore, in the grabbing process of images, there is a loss of information, due to the distortion of the camera, noise or blur. So, one can assume that capturing consists of transforming a high-resolution to a low resolution image and can be written as follows :

$$y_k = DB_kM_kx + n_k \tag{9}$$

where y_k is the k^{th} low resolution image of the sequence, x the high resolution image, D the decimating matrix, B_k the blur matrix, M_k the warping matrix representing the motion that occurs during image acquisition and n_k the noise vector.

Most of the SR image reconstruction processes consist of three steps : registration, interpolation and restoration. Registration refers to the estimation of motion. Since the shifts between LR images are not regular, the registered low resolution image, will not always correspond to a uniformly spaced high resolution grid. Thus, non-uniform interpolation is necessary to obtain a regular high resolution image. Finally, image restoration is applied to the up sampled image to remove blurring and noise.

5.3 Super Resolution Method

Usually, in disparity computation process, it is assumed that the disparity range is discretized to one pixel. To improve the sharpness of the results, the idea presented here is to use a disparity range discretized to a half pixel. This means, working with a disparity interval $[d_{min}, d_{max}]$ of size D. We would like to refine the disparity map assuming a disparity interval $[d_{min}, d_{max}]$ of size 2 $\times$ D. Considering this like multiplying the image width by a magnification factor of 2, new pixels appear. Intensity values have then to be assigned to them. A first and obvious idea would be to interpolate the intensities of the neighboring pixels. But it supposes that the texture varies homogeneously. To avoid this false assumption, super resolution image reconstruction technique is used to computed the texture of the new pixels.

The first frame is used as a reference frame and is placed on a regular grid of size $M \times N$. The 3 other LR images are placed taking into account the shifted position of the pixels estimated by the optical flow (see 4). A new column is then inserted between the existing ones, such as the the new grid size equals to $2M \times N$. Finally, the intensity of every new pixel (corresponding to the new column) are computed using a weighted nearest neighbor approach [1]. The cost $c_{p,d}$ is so redefined:

$$c_{p,d} = I_1(x,y) - I_2(x+d,y) \tag{10}$$

with

$$\begin{cases} I(x,y) = I_{LR,0}(x/2,y) & \textbf{if } x \textbf{ is even} \\ I(x,y) = \sum_{k=0}^{2} \pi_k J_k & \textbf{else.} \end{cases}$$

where $I_{LR,t}$ is the LR image of the sequence at time t, $J = \{J_0, J_1, J_2\}$ is the set of the three nearest neighbors of pixel(x,y), on the super resolution grid, among shifted image $I_{LR,t}$, and π_k is the weight inversely proportional to the distance to the pixel(x,y).

6 Results

The size of our low resolution image is 320×240, with a disparity interval size of $D = 11$ pixels. The disparity map reference is computed from 640×480 images. As

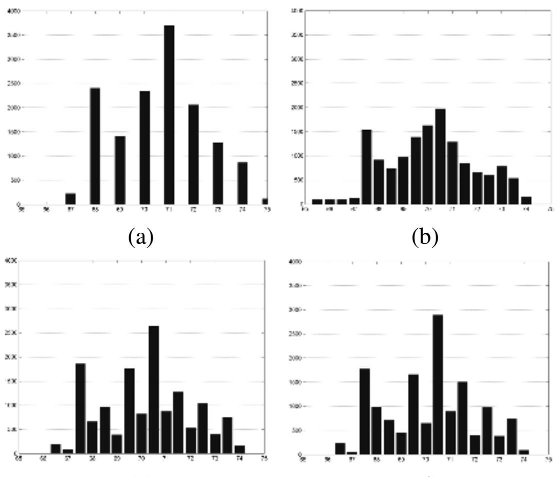


Fig. 3. Distribution of Pixels in function of the disparity. (a) LR, (b) Reference, (c) SRWOF, (d) SROF.

Table 1. Comparisons with reference disparity map. LR: Low Resolution - SRWI: Super Resolution in Width by Interpolation - SRWOF: Super Resolution in Width by Optical Flow - SRI: Super Resolution by Interpolation - SROF: Super Resolution by Optical Flow.

Methods	Distance (L2 norme)	Ratio	# of Differences
LR	8151.5	2.79	10805
SRWI	4735.75	1.62	7516
SRWOF	3685.75	1.26	6350
SRI	4308.75	1.47	6861
SROF	2926.5	1	6057

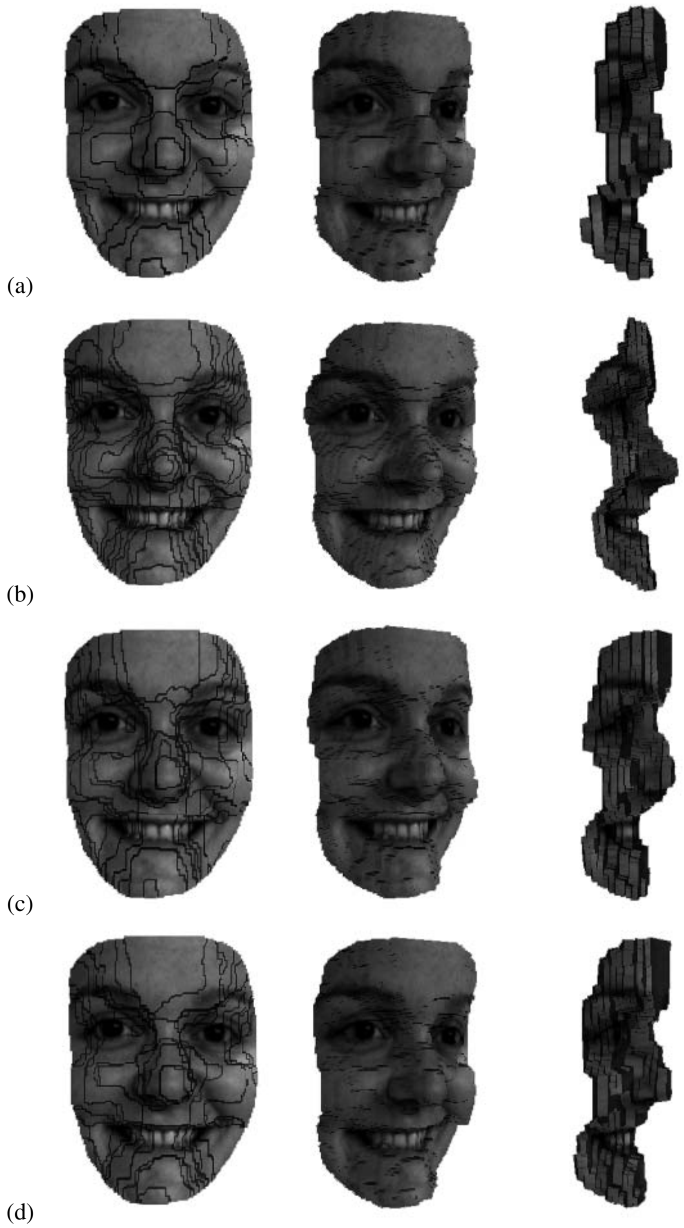


Fig. 4. (a) Low Resolution, (b) Reference, (c) Super Resolution in Width by Optical Flow, (d) Super Resolution by Optical Flow

we use a magnification factor of 2, the new disparity interval size is $2D$. Four couples of subsequent images are used to compute the super resolution. Table(1) shows the comparisons between results of different methods and the reference disparity map.

Obviously super resolution in two dimensions gives the best results in terms of differences with reference disparity map, but super resolution in width gives close similar results. Fig.(3) gives a representation of the pixel distribution in function of the disparity. One can see that super resolution techniques distributions are influenced by the ”low resolution data” and are quite comparable to each other.Finally Fig.(4) shows different 3D reconstructions of the same person, with mapped texture, using the same sequence, with same parameters in graph cut process. The third column puts the sharpness of the super resolution techniques in obviousness.

The results enlarging width or height and width can be discussed. Even if Table(1) gives the super resolution in two dimensions as the best one, the visual results show that super resolution in width, could be at a sufficient fine level. Considering the magnification factors of both super resolutions and the complexity of max flow algorithm (O(Number of Nodes $\times$ Number of Vertices $\times$ Cost of the cut)) one can assume than this new method is a good compromise between resolution and computation time.

7 Discussion

In this paper we have proposed a new method to acquire high resolution depth maps of facial expressions from low resolution images. Our method exploits temporal coherence between images and a powerful reconstruction method driven from combinatorial optimization. The concept of super solution is explored in an implicit form. To this end, reconstruction is reformulated as a max flow/min path problem in a graph where temporal coherence between couples of stereo images is encoded in the construction of such a graph. Classic techniques of optimization are then used to recover the optimal reconstruction map.

Better and direct exploitation of super resolution principles is an straightforward extension of our approach. Once high resolution models have been recovered, reproducing facial animations is a challenging and interesting extension with numerous applications. To this end, registration of expressions as well as autoregressive modeling of transitions between one expression to the next are to be addressed. Real-time performance is also a challenging perspective where the use of graphic processing units could address such a demand.

References

1. M. Alam, J. Bognar, R. Hardie, and B. Yasuda. Infrared Image Registration and High-Resolution Reconstruction Using Multiple Translationally Shifted Aliased Video Frames. In *IEEE Transactions on instrumentation and measurement*, volume 49, pages 185–203, 2000.
2. V. Blanz and T. Vetter. A Morphable Model for the Synthesis of 3d Faces. In *SIGGRAPH*, page 187194, 1999.
3. Y. Boykov and V. Kolmogorov. An Experimental Comparison of Min-CutMax-Flow Algorithms for Energy Minimization in Vision. *Pattern Analysis and Machine Intelligence*, 26:1124–1137, 2004.
4. Y. Boykov, O. Veksler, and R. Zabih. Efficient Approximate Energy Minimization via Graph Cuts. *Pattern Analysis and Machine Intelligence*, 20(12):1222–1239, 2001.

5. T. Brox, A. Bruhn, N. Papenber, and J. Weickert. High Accuracy Optical Flow Estimation based on a Theory for Warping. In *ECCV*, 2004.
6. D. DeCarlo and D. Metaxas. Optical Flow Constraints on Deformable Models with Applications to Face Tracking. In *International Journal of Computer Vision*, 2000.
7. O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, 1993.
8. O. Faugeras and R. Keriven. Complete Dense Stereovision Using Level Set Methods. In *European Conference for Computer Vision*, pages 379–393, 1998.
9. S. Ilic and P. Fua. Generic Deformable Implicit Mesh Models for Automated Reconstruction. In *ICCV workshop on Higher-Level Knowledge in 3D Modelling and Motion Analysis*, 2003.
10. V. Kolmogorov and R. Zabih. Multi-camera Scene Reconstruction via Graph Cuts. In *ECCV*, volume 3, pages 82–96, 2002.
11. K. Kutulakos and S. Seitz. A Theory of Shape by Space Carving. *International Journal of Computer Vision*, 38:199–218, 2000.
12. Z. Liu, Z. , Zhang, C. Jacobs, and M. Cohen. Rapid Modeling of Animated Faces from Video. In *International Conference on Visual Computing*, pages 58–67, 2005.
13. N. Magnenat-Thalmann, E. Primeau, and D. Thalmann. Abstract Muscle Action Procedures for Human Face Animation. *The Visual Computer*, 3, 1987.
14. S. Park, M. Park, and M. Kang. Super-Resolution Image Reconstruction : A Technical Overview. In *IEEE Signal Processing Magazine*, pages 21–36, 2003.
15. O. Veksler. Stereo Correspondence by Dynamic Programming on a Tree. In *Computer Vision and Pattern Recognition*, 2005.
16. C.L.Y. Wang and D.R. Forsey. Langwidere : A New Facial Animation System. In *Computer Animation*, pages 59–68, 1994.

Web-Based Interface for the Visualization of Microarray Data

B. Vanteru, J. Shaik, and M. Yeasin

Electrical and Computer Engineering,
Computer Vision, Pattern and Image Analysis Lab,
University of Memphis, Memphis, TN- 38152, USA
{bcvanter, jshaik, myeasin}@memphis.edu

Abstract. This paper presents the design and development of a web-based interface for the visualization of high dimensional data such as microarray data. A co-ordinate based method, namely, 3D Star Coordinate (3SC) projection technique is used for the visualization. The proposed web-based interface enables the user to choose an existing dataset from the database or upload a dataset and visualize the best possible projection of the data on an applet running on the client web browser. The proposed projection algorithm runs in Matlab at the server side for faster computation and using Java Servlets the results are delivered to the client machine.

Keywords: Data Projection, Web-based Interface, Coordinate-based & Axis-based Visualization, Dimensionality Reduction and Data Integration.

1 Introduction

This paper proposes a web-based interface for the visualization of high dimensional data. Visualization is an integrative part of discovering knowledge from high dimensional data such as microarrays. In general areas of computational biology and bioinformatics, the need is even greater as there are hardly any visualization tools available online to understand the underlying distribution of the complex data. Such a knowledge can help in choosing the appropriate algorithm for the data analysis. The projected results of complex data offer good insight to the data before analysis and also help analyze the result and relate it with possible biological concepts.

A simple yet powerful web-based user friendly interface with easy interpretation of the results is the focus of this paper. A web-based application is a server side application where the computations are performed at the server side and the results are delivered to the applet running on the client machine. Advantages of such a method include but not limited to cross-platform usage and consistency in performance irrespective of the power and memory limitations of the client machine etc. On the other hand, the stand-alone application performance depends on factors such as complexity of the data, performance of the user machine etc. Many of the visualization tools available today [1] are stand alone. The algorithms on which these interfaces are built might be based on same hypothesis with variations customized for a particular data. The web based tools available are standard and may not be suitable for all the applications[2]. Star coordinate based methods are found to be excellent tools for visualization of high dimensional data. This 3D star coordinate projection has been the area of interest for the researchers dealing with high dimensional data recently [3, 4] (Appendix A). There are no web-based tools for the visualization of high dimensional data using coordinate based methods viz. 3D star coordinate projection.

This paper presents a web-based application for 3D star coordinate based projection. This application may be accessible via. [5]. The visualization algorithm was initially built for knowledge discovery in microarray data and tested for its applicability for databases of various other fields. For example, for the microarray data dealing with cancer and normal cells, we know that cancer cells contain substantial number of genetic abnormalities when compared to normal cells. These experiments are performed to find the genotypes which are responsible for a particular phenotype (cancer cell in this case). After those genes are identified based on some criterion [6-8], it is required that a visual interpretation of the separation of phenotypes is offered for confirmation. Visualization algorithms are run using the selected genes as features to see if they offer separation between the different phenotypes. If the separation of phenotypes is visually obvious then most or all of these genotypes might be involved in the formation of a particular phenotype. Else, the genotype selection algorithm might have to be run with different parameters. 3D star coordinate web application has been run on microarray data. Empirical analysis shows that the application is well suited for the visualization of high dimensional data. This interface is web-based and hence executes on a web browser. This makes Java applet technology an obvious choice since applets can easily be sent over the internet to the client and since they are written in java, they can communicate with any java program running at the server. They also provide security in the form that the data sent by the user is securely sent over the network by using encryption.

The rest of the paper is organized as follows: Section II discusses the works related to web-based visualization techniques with brief discussion on visualization techniques themselves. Section III discusses the implementation details of web-based interface. Section IV discusses the features of the interface and Section V presents some results and discussions related to the development of the interface. Finally, Section VI offers some concluding remarks and Future works.

2 Related Works

A good web-based interface conveys the intended message with less complexity**.** The idea behind web-based interface is two folds: 1) offer the results to the client requesting access in easily interpretable form and 2) restrict access to the code. A variety of visualization techniques have been developed for knowledge discovery in databases [3-5, 9-18]. Although some of the visualization methods have been well established, the web applications of some of these algorithms have not been developed. Some of the well established and most visited web applications include but not limited to PCA [2, 15, 19], LDA [2, 20], clustering based methods [2, 21-23] etc. No attempt has been made to build web applications for axis based methods. One possible explanation for this could be that these methods are relatively new and much research is under progress [3, 4, 14]. This paper presents a web based 3D star coordinate visualization tool for high dimensional data such as microarray data [5]. Although some of the techniques are well developed, it is the ability of the visualization algorithm to offer results in real time that forms basis for the development of the web-based interface. The basic requirements for the visualization algorithm underlying the web-based interface include but not limited to a) Ability to handle large number of data points, b) Offer multiple views for knowledge discovery, c) Ability to handle growing databases, etc.

Visualization techniques may be broadly categorized into three groups i) positional ii) temporal and iii) retinal. Positional techniques can be one, two or three dimensional [9, 13, 14, 20, 24-26]. 2D plots may be categorized into scatter plots, bar plots, line plots, pseudo color images, arrow plots and contour plots. The 3D plots may be divided into scatter plots, surface plots, flow ribbons, particles, arrow plots and volumes. The only temporal technique is animation. Retinal techniques may be set of retinal properties of marks such as texture, orientation, size, shape, and color. A good visualization technique defines the success of the web-based application. Based on the expected information retrieval and type of result expected, a particular algorithm may be selected. 3D star coordinate algorithm's ability to offer faster results and handle growing databases makes it favorable for microarray datamining.

3 Implementation Details of the Application

This paper presents a web-based application for 3D star coordinate based projection. This application may be accessible via. [5]. The visualization algorithm was initially built for knowledge discovery in microarray data and is tested for its applicability for databases of various other fields. Empirical analysis shows that the application is well suited for the visualization of high dimensional data. This application has two major modules, one module runs at the client side, which is a Java Applet and the other module runs at the server side called Servlet. The communication between two modules is established using Java Sockets. This Web based application enables the user to either upload his own data which he intends to gain insight into or select a dataset already existing in the database. After selecting a data for visualization, the user activates the execute function. The request to run the algorithm on the specified dataset is sent to the servlet running on the server side. Upon receiving the request, the servlet activates the Matlab system to execute the algorithm on the specified dataset. The results of the computation are stored on the server machine as binary data and passed on to the client machine as per the navigation request and displayed as a plot. The user navigates through different projections (plots) until he discovers underlying pattern of the data. In order to provide enhanced security, this interface deletes the data (if data is uploaded) after the results are offered to the client. The added advantage of this process is that multiple users may access the site at the same time as Matlab can create multiple sessions.

This web application has three basic components viz. i) Initialization ii) Action Selection and iii) Termination.

i) **Initialization:** In this stage, the applet loads onto the web browser along with its components and waits for the user action.
ii) **Action Selection:** The action taken by the user defines this stage. The user action has two initial selections, select the existing database or upload the

data he intends to gain knowledge about. If the user uploads his data, a Java socket connection is established with the server and data is transferred to the server using serializable objects. Upon selection of data, the application waits for the user select one of the three existing algorithms (one manual and two automated) and press the execute button. When the execute action is performed by the user, a server program is invoked which executes the algorithm with given parameters in matlab. Now the interface waits until the algorithm is executed and the results are ready. Then the interface gets the results from the server and displays the results. The user can then navigate through different projections by using the next and previous buttons. Clicking on the plot and rotating it to the required elevational and azimuthal angles for enhanced knowledge discovery may obtain different 3D views. The location of a data point in 3D space may be obtained by placing mouse over that particular datapoint. The zoom in and zoom out features may be used accordingly for concentrating on a particular area. If the user is satisfied with a particular projection, the coordinates corresponding to that projection may be obtained using download feature.

iii) **Termination:** The interface is terminated when the user closes his web-browser. The following events occur when the user terminates the interface.

a) All the variables used by the interface are released by calling the garbage collector.

b) A ‘DeleteServlet’ program in the server is called which will delete all the results stored including the uploaded data.

4 Web-Based Interface

The web-based interface consists of 7 fields (dropdown menu/upload, Algorithm selection, Execute, zoom in/ zoom out, download, Next/Previous and display) fields respectively. The functionality of each field is described here:

Dropdown Menu/ upload: This field enables the user to choose any of the datasets already available in the database or upload (The current acceptable file format is Microsoft excel. The future application will accept text-delimited format. The file should contain data points along rows and the dimensions that need to be reduced along columns) his dataset and view the output of the proposed visualization algorithm. The datasets already in the database are Automobile [27], Iris dataset [28], Simulated Gaussian [20], Leukemia microarray dataset [29], Petroleum dataset [30], and Gastric Cancer microarray dataset [31].

Algorithm Selection: This feature enables the user to select one of the existing algorithms. One manual and two automated algorithms are proposed.

Execute: This button initiates the implementation of 3D star coordinate projection algorithm at the server side and the results are displayed in the display field on the client web browser. The user has the flexibility of choosing one of the existing datasets or uploads his dataset using the upload button**.**

Next and Previous: These fields offer access to available projections. For each combination of axes, 3D star coordinate projection algorithm provides a new projection.

One or more such combinations provide insight into the structure of the data. Visual clusters may be obtained for such combinations.

Zoom in/ Zoom out: This feature enables the user to zoom in and out of the area of interest in the 3D space.

Download: This feature enables the user to download the data in projected space corresponding to the currently viewed projection.

5 Results and Discussions

The screen shot of the proposed web-based interface is shown in the Figure 1. The Figure 1 shows the results of one possible projection for the Leukemia microarray dataset. From the Figure 1 it is evident that the dataset has two distinct classes (phenotypes) as identified by the 3D star coordinate algorithm. The boundaries between the classes are well defined and crisp. Highly differentially expressed 50 genes are found using [6-8]. These genotypes are considered for projection of two tissue cases (phenotypes). From Figure 1 display module, it is clearly seen that normal and pathological tissue samples are well separated using differentially expressed genes.

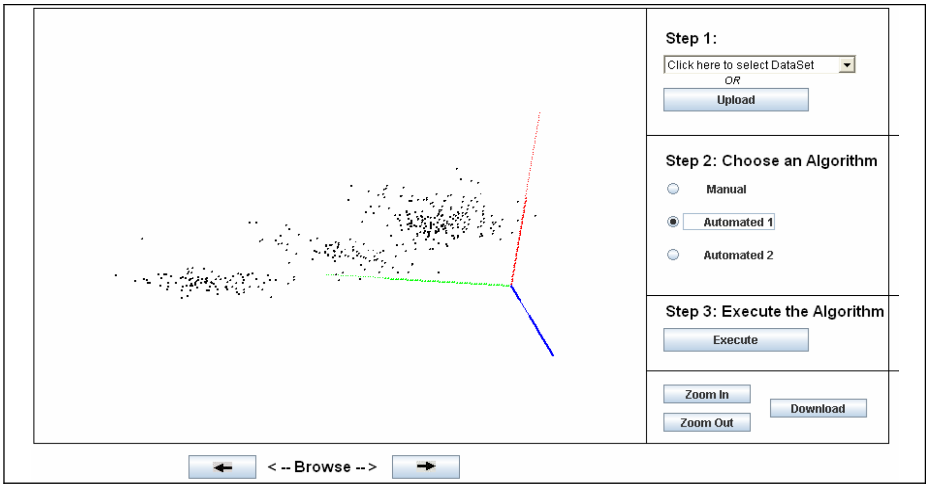


Fig. 1. Proposed Web Interface for visualizing high dimensional data using the 3D star coordinate Projection

The 3D star coordinate projection algorithm is implemented in Matlab. The web-based application was initiated based on the assumption that communication between Java and Matlab would be accomplished. Since the methods to communicate between Java and C have been well addressed, our initially thought was that Matlab script would be first converted into 'C' and then communication would be established between Java and C using Java Native Interface (JNI). But, it was found through the experimentation that the Matlab compiler did not produce efficient code that may be used for real time. This problem was tackled by establishing a direct connection between Java and Matlab using Runtime class in Java.lang package of Java standard Development kit (JSDK). The Runtime class is used to initiate a new Matlab session.

Runtime class can only initiate Matlab but cannot pass on commands to Matlab. To tackle this problem, the following procedure has been adopted. The IP address of the machine requesting access is written on to a text file 'IPaddress.txt', a new text file is created with IP address as name and the name of the dataset that needs to be executed is written on to it. Upon pressing the execute button at the client side, Matlab reads the IP address of the machine requesting access from 'IPaddress.txt'. Then it searches for the file with the name IP address on the local disk and reads it. As it contains the information about what file the visualization algorithm needs to be run on, the appropriate file is read and 3D star coordinate projection algorithm is initiated. The problem now is to send the results of computation such as plots to the client side using Java. Matlab cannot communicate with Java and hence results of computation may not be passed on directly to client machine while Matlab is running.

This problem has been addressed by saving the images in the local disk and making it accessible to Java. The Java Applet continuously searches for the images until it finds one, and then displays it on the client machine. Images corresponding to the other projections may be accessed appropriately by browsing using arrow buttons on the applet.

6 Conclusions

This paper presents a web-based application for visualization of high dimensional data. This tool is currently used for very high dimensional datasets such as microarray data. The tool has been tested extensively using data belonging to diverse fields with encouraging results. Interplay between Matlab and Java is performed for the functioning of the application. The user may play with the tool by using the existing data in the database or gain insight into his data by trying the 3D star coordinate algorithm using this application in real time. The multiple views, multiple projections and zoom in and zoom out features offer better knowledge discovery. The issues associated with the communication between Java and Matlab have been well discussed.

The current acceptable format for this applet is Microsoft excel. The future version will be able to handle variety of file formats. The algorithm running on the application is not the automated version proposed in [4]. The automated enhanced algorithm selects the projections that preserve the distance between the data points in the high dimensional and projected space, respectively.

Acknowledgements

The authors acknowledge the Herff Fellowship and faculty start-up grants from the University of Memphis for partially funding of this research.

References

[1] S. T. Teoh and K. L. Ma, "PaintingClass: Interactive construction, Visualization and Exploration of Decision Trees," *In Proceedings of 9th ACM SIGKDD*, pp. 667-672, 2003.

[2] Stanford-biomedical-Informatics, "Classification of Expression Arrays(CLEAVER) http://classify.stanford.edu."

[3] J. Shaik, C. R. Kothari, D. J. Russomanno, and M. Yeasin, "3D Visualization of Relation Clusters from OWL Ontologies," *Proceedings of SWWS'06- The internatiinal conference on Semantic Web and Web Services*, 2006.

[4] J. Shaik and M. Yeasin, "Visualization of High Dimensional Data using an Automated 3D Star Co-ordinate System," *Proceedings of IEEE IJCNN'06, Vancouver, Canada.*, 2006.

[5] Computer_Vision_Pattern_and_Image_Analysis_Lab, "3D Star Coordinate Based Web Application http://www.ee.memphis.edu/people/faculty/myeasin/CVPIA/Demo/index.htm." Memphis, Tennessee., 2005-2006.

[6] J. Shaik and M. Yeasin, "Adaptive Ranking and Selection of Differentially Expressed Genes from Microarray Data," *WSEAS transactions on Biology and Biomedicine*, vol. 3, pp. 125-133, 2006.

[7] J. Shaik and M. Yeasin, "A Progressive Framework for Two-Way Clustering Using Adaptive Subspace Iteration for Functionally Classifying Genes," *Proceedings of IEEE IJCNN'06, Vancouver, Canada.*, 2006.

[8] J. Shaik and M. Yeasin, "Performance Evaluation of Subspace-based Algorithm in Selecting differentially Expressed Genes and Classification of Tissue Types from Microarray Data," *Proceedings of IEEE IJCNN'06, Vancouver, Canada.*, 2006.

[9] J. Bertin, *Graphics and Graphic Information Processing*: Walter de Gruyer & Co, Berlin, 1981.

[10] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*: Plenum Press, New York, 1981.

[11] H. Chernoff, "The Use of Faces to Represent Points in k-Dimensional Space Graphically," *Journal of American Statistical Association*, vol. 68, pp. 361-368, 1971.

[12] P. E. Hoffman, G. G. Grinstein, K. Marx, and I. Grosse, "DNA Visual and Analytic Datamining," *IEEE visualization'97*, pp. 437-441, 1997.

[13] A. Inselberg, "n-dimensional graphics, part I-lines and hyperplanes," IBM Scientific center, Los Angeles(CA) 1981.

[14] E. Kandogan, "Star Coordinates: A multi-dimensional Visualization Technique with Uniform treatment of Dimensions," *Proceedings of IEEE information Visualization Symposium*, 2000.

[15] National_Institute_of_Allergy_and_Infectious_Disease, "Database for annotation and Visualization and Integrated discovery (David) http://david.abcc.ncifcrf.gov." Maryland, US.

[16] K. A. Olsen, R. R. Korfhage, K. M. Sochats, M. B. Spring, and J. G. Williams, "Visualization of Document Collection: The VIBE System," *Information Processing and Management*, vol. 29, pp. 69-81, 1993.

[17] R. M. Pickett and G. G. Grinstein, "Iconographic Displays for Visualizing Multidimensional Data," *IEEE conference on Systems, Man and Cybernetics*, vol. 1, pp. 514-519, 1988.

[18] D. Ridder, O. Kouropteva, O. Okun, M. Pietikainen, and R. P. W. Duin, "Supervised Locally Linear embedding," *International Conference on Artificial Neural Networks*, pp. 333-341, 2003.

[19] Integromics, "Gene Engine http://www.engene.cnb.uam.es." Madrid, Spain.

[20] C. Unsalan and A. Ercil, "Shapes of features and a modified measure for Linear Discriminant Analysis," *OCPR00*, vol. 2, pp. 410-413, 2000.

[21] European_Bioinformatics_Institute, "Expression Profiler http://www.ebi.ac.uk/ expressionprofiler/." Cambridge, UK.
[22] National_Center_for_ontological_investigations, "Gene Expression Pattern Analysis Suite (GEPAS) http://bioinfo.cnio.es." Madrid, Spain.
[23] X-mine_Inc, "X-Miner (http://www.x-mine.com)."
[24] D. Asimov, "The grand tour: a tool for viewing multidimensional data," *SIAM Journal on Scientific and Statistical Computing*, vol. 6, pp. 128-143, 1985.
[25] J. Beddow, "Shape coding of multidimensional data on microcomputer display," presented at Proceedings of 1st international conference on Visualization 90, Sanfransisco, california, 1990.
[26] H. Chernoff, "The Use of Faces to Represent Points in k-Dimensional Space Graphically," *Journal of American Statistical Association*, vol. 68, pp. 361-368.
[27] R. Quinlan, "Combining Instance-Based and Model-Based Learning," *In proceedings of tenth International conference of Machine Learning*, pp. 236-243, 1993.
[28] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annual Eugenics*, vol. 7, pp. 179-188, 1936.
[29] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, pp. 531-537, 1999.
[30] S. Makridakis, S. C. Wheelwright, and R. J. Hyndman, *Forecasting:Methods and Applications*: John Wiley and Sons, 1998.
[31] X. Chen, S. Y. Leung, S. T. Yeuen, K. M. Chu, J. Ji, R. Li, A. S. Y. Chan, S. Law, O. G. Troyanskaya, J. Wong, S. So, D. Botstein, and P. O. Brown, "Variation in Gene Expression Patterns in Human Gastric Cancers," *Mol Bio Cell*, vol. 14, pp. 3208-3215, 2003.

Appendix A: 3D Star Coordinate Algorithm

The schematic diagram of the 3D star coordinate [3, 4] is shown in Figure 2. It is assumed that all the dimensions are radiating from the center of a hypothetical sphere at random angles in 3D space and are of random length. The attributes of individual data points are scaled and rotated accordingly. For a given data point, each of its attributes (points along all dimensions of that data point) is multiplied with the component along a particular direction (x, y, z). Individual contributions are summed up along that direction to form a projected space as shown in equation 6. Various stages and steps involved in the formation of 3D star co-ordinate system are succinctly summarized as follows:

Step 1: Initialization

1. Arrange the co-ordinate axes on the sphere with all vectors radiating into the 3D space.
2. The angles between the vectors are random. Each of the dimensions is assumed to be along these vectors.
3. Star co-ordinate system may be mapped to Cartesian co-ordinate system by defining a point in the 3D space representing the origin (O_x, O_y, O_z) and 'n' 3 dimensional vectors $A_n = \langle a_1, a_2, a_3, ...a_n \rangle$ representing the axes.

4. The lengths of the vectors are random. The lengths may vary between one unit vector and 5 times a unit vector.

Step 2: Computation of Projection
The projection of the data point in n-dimensions to 3 dimensions is obtained by summing up the unit vectors (u_x, u_y, u_z) on each co-ordinate weighted by their respective data element.

Here,

$$u_{xy} = \sqrt{\left(u_x^{\,2} + u_y^{\,2}\right)}, \tag{1}$$

$$u = \sqrt{u_{xy}^{\,2} + u_z^{\,2}}\,, \tag{2}$$

$$u_x = u \sin\phi \cos\theta\,, \tag{3}$$

$$u_y = u \sin\phi \sin\theta\,, \tag{4}$$

$$u_z = u \cos\phi\,. \tag{5}$$

Let '*N*' be the number of data points and '*n*' be the dimension of each feature that needs to be visualized. The data matrix '*D*' is of dimension *N x n* and its elements d_{ij} representing the components of *n* dimensional data points. Where, O_x, O_y and O_z are the coordinates defining present origin of the system. The notation $\min(D(:,i))$ stands for minimum value in all rows and i^{th} column. X_P, Y_P, Z_P are the projections along (x, y, z) axes, respectively. The vectors in $'n'$ dimensional space are projected into 3D space given by $P\,(X,Y,Z)$. Depending on the number of dimensions involved in the dataset, which is $'n'$ in this case, $'n'$ such vectors exist in the 3D space whose individual contributions are taken as shown by Equation 6 to project into 3D space.

```
for j = 1 : N,
    $X_p = O_x + \sum_{i=1}^{n} u_{xi}(d_{j,i} - \min(D(:,i))),$
    $Y_p = O_y + \sum_{i=1}^{n} u_{yi}(d_{j,i} - \min(D(:,i))),$        (6)
    $Z_p = O_z + \sum_{i=1}^{n} u_{zi}(d_{j,i} - \min(D(:,i))),$
    $P_j(X,Y,Z) = (X_P, Y_P, Z_P).$
end
```

It is now easy to visualize various dimensions radiating at random angles in a pseudo sphere, more than one such combination providing insight into the underlying distribution of the data while providing relationship between various attributes involved.

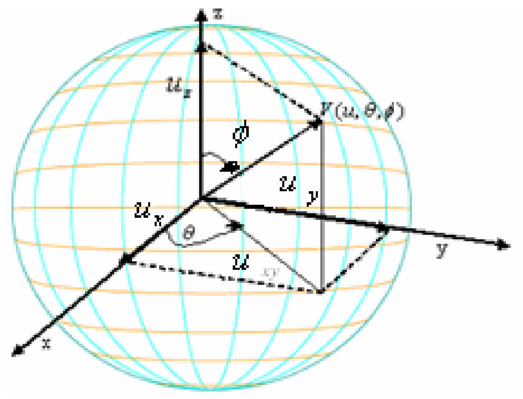


Fig. 2. 3D Star Co-ordinate System

3D and Texture Modelling of Precolombian Objects

Jorge Hernández and Flavio Prieto

Universidad Nacional de Colombia sede Manizales,
Carrera 27 N 64-60, Manizales, Caldas, Colombia
{jehernandezl, faprietoo}@unal.edu.co
http://www.unal.edu.co

Abstract. In this paper we present a 3D and texture modelling for Precolombian objects. Our experimental setup consists of a no contact 3D digitizer for range image acquisition and a camera CCD of high resolution to acquire the intensity images of the target object. The mapping texture process is described as a parameterization function with a range image; we estimate the camera orientation from calibration techniques, utilizing pattern calibration before data acquisition. We describe a texturized mapping strategy based on multi-view to adequately address photography related problems such as inhomogeneous lighting, highlights and occlusion. For each triangle in the model, the optimal image is selected from source images with an average weighted the scalar product between the image normal vector and triangle normal vector. Finally, we show the highly detailed texture models of Precolombian objects.

1 Introduction

In recent years, computer vision and computer graphics are two fields that are gradually merging through the advent of Virtual Reality (VR). More and more applications use computer vision methods to build accurate models of objects/scenes from actual data [1]. Using VR, we can display and manipulate a three-dimensional model representation of an existing object's shape. Appearance can play a significant role in the area of preserving historical remains, or cultural heritage [2]. The importance of cultural heritage is well defined by UNESCO, which is an organization that represents and supports the integrity of the cultural heritage.

Different works have been presented according to the culture [3]. The most important efforts in this area were the "Digital Michelangelo Project" [4] and the "Piet Rondanini Project" [5], both concerning Michelangelo's artworks. Equally, the campaign to acquired the statues of Donatello and Giovanni Pisano jointly performed by the Visual Information Technology Group of the NRC of Ottawa, Canada [6]. Another project that works on sculptures is the modelling of the relics of the Museum of Quin Shihuang Terra Cotta Warriors and Horses [7] and of the 15 m high Kamakura's Buddha [8] and Koumokuten [9]. Precolombian objects, such as ceramics, vases, jewelery, etc, with an immense quantity of designs are an important part of the American cultures, and for this reason, they are our studio case. Several related works are carried out on the cultural conservation of scenes, environments and monuments of great size. The Precolombian objects, however, are characterized as relatively small and highly textured, they have severeal modelling difficulties due to many causes such as their shape, typically very articulated and with high auto-occlusion; size and the need to acquire color, texture appearance or reflectance information [6]. In regards to cultural heritage and cultural conservation, the first contribution of this paper concerns Precolombian 3D modelling and a texture mapping process of high resolution and high accuracy. Previously, a number of techniques dealing with the reconstruction of the texture have been developed. Sato et al. [10] faithfully reconstruct the reflectance properties of real world objects from photographs. They acquire the geometry and texture information at the same time, using the same sensor. Pulli in [11], presents a complete system that uses a stereo camera with active lighting to scan the object surface geometry and color as visible from one point of view. Many previous algorithms tried to find the camera transformation by minimizing the error between the contour or shape found in the image and the contour or shape of the projected 3D model [12, 13, 14]. The second contribution of this paper, with respect to texture mapping, concerns the strategy based on multi-view to adequately address photography related problems such as inhomogeneous lighting, highlights and occlusion; also, for each triangle in the model, the optimal texel image is selected from source texel images with an average. The last contribution is the texture model evaluation which approaches using different points of views and multiple views.

Finally, our goal is to create an accurate geometric and photometric representation of the Precolombian objects by means of integrating range and image measurements. The geometry of an object is captured using range sensing technology whereas the information texture is captured by means of cameras (Section 2.1). We have developed a calibrated system, which produces a geometric and photometric correct high resolution and accurate 3D model representation (Section 2.2). The processes of multiple views are described in the Section 2.2. The results and their evaluation are showed in the Section 3.

2 Texture Object Modelling Procedure

2.1 Acquisition System

Our experimental set up consists of the no contact 3D digitizer Minolta VIVID9i for range images acquisition and the camera CCD SONY DSC 717 to acquire the intensity images of the target object (Figure. 1). Although the 3D digitizer can produce the color images as well as the 3D geometry, we utilized the digital camera to get high quality images. The resolution of acquired images is 2560×1920.

The relative position and direction between range sensor and digital camera is necessary to map the image of the digital camera onto the 3D geometric model measured by range sensor. Several researches have been done to estimate the relationship automatically [12, 13, 14, 15]. Nevertheless, the calibration before scanning approach is the most effective method for applications with high accuracy, and we used this approach. The camera model utilized to know the relationship between the intensity image and range image is the pin-hole camera model. The model camera has two parameters: extrinsic and intrinsic. The relative position and direction between range sensor and digital camera is necessary to map the image of the digital camera onto the 3D geometric model

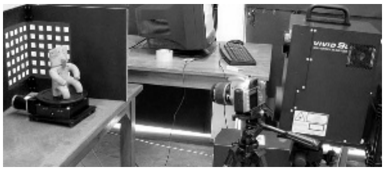
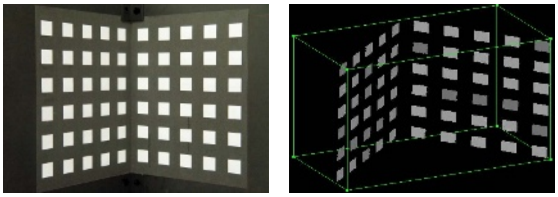


Fig. 1. Data acquisition setup



(a) Intensity Image (b) Range Image

Fig. 2. Pattern Calibration with both sensors

measured by range sensor. Calibration techniques usually make use of a 3D known object which is called the calibration pattern. Camera calibration consists of the estimation of a model for an un-calibrated camera. The objective is to find the external parameters (position and orientation relative to a world co-ordinate system), and the internal parameters of the camera (principal point or image center, focal length and distortion coefficients). One of the most used camera calibration techniques is the one proposed by Tsai [16]. We now designed a pattern calibration, which can easily measure the corresponding points between intensity image and range image. The 3D digitizer employs an option by setting up the laser intensity; when the laser intensity is low ($< 15/255$), the digitizer can't acquire the black part's objects. We use this option to build a black pattern calibration with white squares (Figure. 2). The 3D digitizer produces 3D geometry (Figure. 2(b)) and color image of the object (Figure. 2(a)) where the relationship between 3D geometry and image is known.

With the two types of acquired images we proceed to calculate automatically the couples of points, using the centers of mass of each square. On range image, we segmented all regions and calculated the mean of the points of each region. In the image of intensity, we carried out the perspective correction, after we segmented the white squares and calculated the mean of the pixels. Finally, when we had the correspondences, we used the calibration algorithm *The Gold Standard* [17], and obtained the camera matrix P (camera parameters).

2.2 Mapping Texture

Texture mapping plays a very important role in the modelling of objects of the cultural heritage. This process is divided into two steps, texture mapping a single range image and texture mapping a multi view.

Mapping Texture a Single Range Image: At this moment, when the camera parameters were known, we used the matrix camera, as parameterization function (Eq. 1), to texture map onto each range image.

$$f(x,y,z) = \begin{cases} u = \frac{p_{00}x+p_{01}y+p_{02}z+p_{03}}{p_{20}x+p_{21}y+p_{22}z+p_{23}} \\ v = \frac{p_{10}x+p_{11}y+p_{12}z+p_{13}}{p_{20}x+p_{21}y+p_{22}z+p_{23}} \end{cases} \tag{1}$$

where, $p_{i,j}$ are elements of camera matrix $(P_{3\times 4})$.

Assigning texture coordinates to a 3D mesh (x, y, z), which can be regarded as a parameterization of the mesh surface, where each 3D vertex is assigned to a 2D parameter value (u, v) and each point on the mesh surface is parameterized by the appropriate convex combination of the parameter values of the vertices of the surface triangle in which it resides [18]. Care must be taken so that the parameterization is legal, i.e. that no two points on the mesh surface are mapped as the same point in the 2D parameter domain. The problem of legal parameterization is present in our acquisition system, because the sensor camera and sensor range aren't aligned completely; the camera is on the left side from the position of the digitizer. To solve this problem, we employed the Z-buffer algorithm [19]. A Z-buffer image keeps each pixel location at the depth value of the surface point nearest to the corresponding image plane and falls onto that pixel when projected. We used the projection of each triangle surface onto the image. Comparing the stored depth value with the actual distance of the triangle to the image plane can determine if there is another surface part between the triangle and the image plane.

Mapping Texture in Multi-views: An object of the real world needs multi acquisition, to be completely reconstructed. After we had all the range images with their parameterization function and their texture, we have to register and integrate in a single mesh. The registration estimate of the rotation and translation between two 3D views method is used in this work to find the transformation (T_i) with the ICP algorithm [20]. After, the process of registration, we merge all views using PET (Polygon Editing Tool) software. This translates into the process of Figure 3.

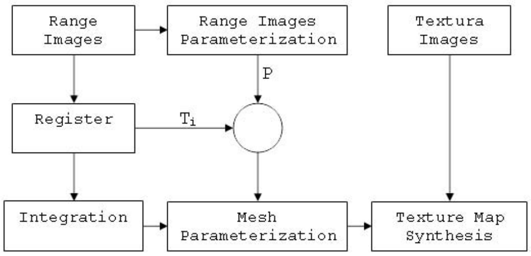


Fig. 3. Process Multi View

The mesh parameterization function involves the camera matrix (P) of surface parameterization function and the transformation matrix (T_i) in the registration process (Eq. 2, Eq. 3)

$$\begin{bmatrix} \widehat{x} \\ \widehat{y} \\ \widehat{z} \end{bmatrix} = (T_i)^{-1} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2}$$

$$f(\widehat{x}, \widehat{y}, \widehat{z}) = \begin{cases} u = \frac{p_{k00}\widehat{x}+p_{k01}\widehat{y}+p_{k02}\widehat{z}+p_{k03}}{p_{k20}\widehat{x}+p_{k21}\widehat{y}+p_{k22}\widehat{z}+p_{k23}} \\ v = \frac{p_{k10}\widehat{x}+p_{k11}\widehat{y}+p_{k12}\widehat{z}+p_{k13}}{p_{k20}\widehat{x}+p_{k21}\widehat{y}+p_{k22}\widehat{z}+p_{k23}} \end{cases} \tag{3}$$

In the multi views process, the texture information can be redundant and transplanted, because some of the surface triangles are visible from two or more camera positions. In the step of texture map synthesis, we employed texel blending [21], whose weights are determined by average dot vector normal; the texel map is the texel from each of the source images with the scalar product between the image texel image camera vector and the triangle normal vector (Eq. 4).

$$Texel_{Final} = \frac{w_1 Texel_1 + w_2 Texel_2 + \cdots + w_{kv} Texel_{kv}}{w_1 + w_2 + \cdots + w_{kv}} \tag{4}$$

where, w_i is the absolute value of the dot among the triangle normal vector and the camera normal vector and kw is the number of visible cameras for the triangle.

Another stage in the step of texture map synthesis is to solve problems which can make an image invalid as a texture source or related problems such as inhomogeneous lighting and highlights. We solve these problems using color by correlation for color constancy [22]. The color constancy considers the problem of illuminate estimation: how, given an image of a scene recorded under an unknown light, we can recover an estimate of that light. Obtaining such an estimate is a central part to solving the color constancy problem of recovering an illuminate independent representation of the reflectances in a scene (Figure 4). Figure 4(a) shows the original sequence of images where the discontinuity is observed among them. After applying the algorithm the discontinuity decreases.

3 Experimental Results and Evaluation

To evaluate the quality of the representation of the texture model, approaches of points of views (shape and contour) and of multiple views (texel correlation) were used.

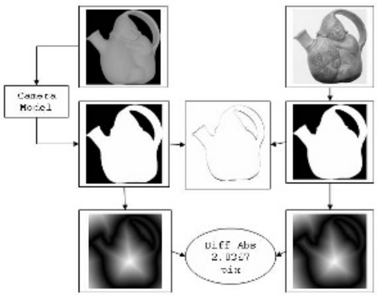
3.1 Points of View Evaluation

This evaluation metric is applied to the models of objects which have very defined shape and contours. We project the model using the camera model or the parameterization function. The resulting shape is compared to the shape of the segmented texture with the comparison being carried out with the absolute difference in the two shapes. After determining each one of the shapes, we calculated the contours followed by a transformation of distance [23]. When carrying out the difference of the transformations of distance of the contours, we calculated the distance between each one of the image contours. The closer the metric values are to 0, the better the quality of the texture model representation.(i.e. Figure 5(a))

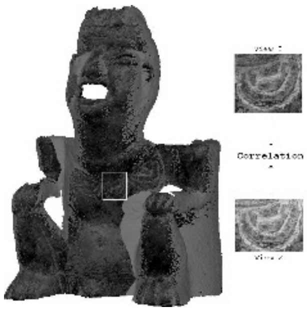
(a) Original sequence of images

(b) Correction sequence of images

Fig. 4. i.e. Color correction for color constancy



(a) Points of View Evaluation



(b) Multiple Views Evaluation

Fig. 5. Results Evaluation

3.2 Multiple Views Evaluation

The texture information can be redundant because there are surface triangles, which have two or more textures visible. The metric evaluation calculates the correlation among the different points of view that belong to one triangle. Metric values fall between 0 and 1; where, if the value is near 0, it means that there is a problem between the textures and the model. Otherwise, if the value is near 1, the possibilities that the texture is registered onto the model correctly are very high.(i.e. Figure 5(b)).

The presented methods were applied to three different Precolombian objects of different cultures:

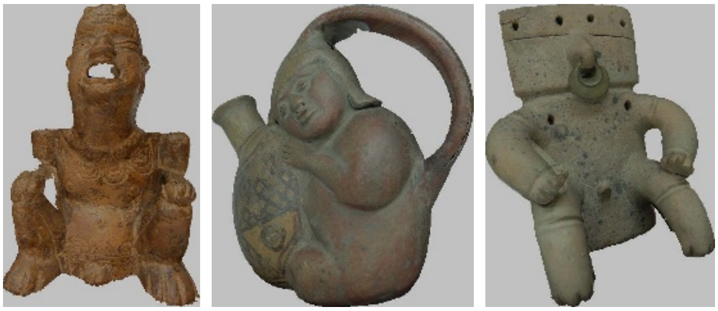
- **Precolombian 1:** anthropomorphous object - *"canastero"*. Archaeological area Calima (Figure 6(a)).
- **Precolombian 2:** anthropomorphous object - Vessel. Archaeological area Ecuadorian (Figure 6(b)).
- **Precolombian 3:** anthropomorphous object - masculine with *"nariguera"*. Altarpiece .Archaeological area Quimbaya (Figure 6(c)).

The photographic images are taken with controlled illumination conditions. The objects are posed on a turntable, for each range image, there is a texture image. We took 14 photographic views, 12 images rotating 30 degrees and top and bottom.

Table 1 shows the metrics evaluation for the three Precolombian objects. The metric values are good, confirming the quality of the texture model representation (Figure 6). Also, we presented the importance of using the illumination correction, because the metric of multi view with illumination correction is higher than the metric of multi view without illumination correction.

Table 1. Texture Model Evaluation

OBJECT	**EVALUATION**			
	Point View		**Multi View**	
	Shape [%]	*Contour [pix]*	*Without Corr. Ilum.*	*With Corr. Ilum.*
Precolombian 1	0.9028	1.78554	0.945302	0.964916
Precolombian 2	0.8259	1.4841	0.964208	0.9842143
Precolombian 3	1.3826	3.2464	0.884496	0.910357



(a) Precolombian 1 (b) Precolombian 2 (c) Precolombian 3

Fig. 6. Texture Model Precolombian Results

4 Discussion and Conclusions

We have described a model texture procedure for Precolombian objects of high resolution and accuracy, acquiring the range image and the texture image at the same time. We got a textured model by mapping the images onto the 3D geometric model using the camera parameters estimated from the calibration employing a pattern calibration. After the calibration process, we can't move the devices' acquisition; when we took the image sequence (range and texture). Also, from this stage the process is totally automatic. The differences in the highlights due to the illumination and to the properties of reflectance of the objects presented in the stage acquisition were reduced using an algorithm of correction of illumination for color constancy for correlation and utilizing an average texel, weighted by dot vector normal which condensed source image frontiers along the surface of the object during the multiple views stage. The metric of multiple views is subject to the reconstruction of the complete model, directly to the registration process. If the algorithm used registration presents a great error in the approach, one cannot wait for a good result in the metric of multiple views. It also presents an additional dependence under the condition of illumination in the acquisition of the different textures. Using the proposed method, experiments of the texture mapping for the 3D geometric models of Precolombian objects were carried out, and the usefulness of the proposed method was verified. The results present show the high level of detail and high degree of realism of the texture models.

Acknowledgement

This work was developed during the master thesis of the first author, who was beneficiary of the *Programa de Becas para Estudiantes Sobresalientes de Posgrado*. The research has been partly supported by project *Modelado de superficies de forma libre empleando técnicas de visin artificial*, it was financed by *Dirección Nacional de Investigación de la Universidad Nacional de Colombia (DINAIN).*

References

1. Denis Laurendeau, N.B., Houde, R.: The mapping of texture on vr polygonal models. Second International Conference on 3-D Imaging and Modelling (3DIM '99) (1999) 332,
2. Guy Godin, J.-Angelo Beraldin, J.T.L.C.M.R.S.E.H.R.B.F.B.P.B.J.D.M.P.: Active optical 3d imaging for heritage applications. Computer Graphics in Art History and Archaeology (2002) 24 – 36
3. Vilbrandt, C., Pasko, G., Pasko, A., Fayolle, P.A., Vilbrandt, T., Goodwin, J.R., Goodwin, J.M., Kunii, T.L.: Cultural Heritage Preservation Using Constructive Shape Modeling. Computer Graphics Forum **23** (2004) 25–41
4. Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., Shade, J., Fulk, D.: The digital michelangelo project: 3D scanning of large statues. In Akeley, K., ed.: Siggraph 2000, Computer Graphics Proceedings. Annual Conference Series, ACM Press / ACM SIGGRAPH / Addison Wesley Longman (2000) 131–144
5. Bernardini, F., Rushmeier, H., Martin, I., Mittleman, J., Taubin, G.: Building a digital model of Michelangelo's Florentine Pietà. IEEE Computer Graphics and Applications **22** (2002) 59–67
6. Marco Andreetto, N.B., Cortelazzo, G.M.: Automatic 3-d modeling of textured cultural heritage objects. IEEE Transactions on Image Processing **13** (2004) 354–369
7. Zheng, J.Y., Zhang, Z.L.: Virtual recovery of excavated relics. IEEE Computer Graphics and Applications (1999) 6–11
8. Katsushi Ikeuchi, Atsushi Nakazawa, K.H.T.O.: Representing cultural heritage in digital forms for vr systems through computer vision techniques. Proceedings of the 17th International Conference on Pattern Recognition (ICPR04) (2004)
9. Unten, H., Ikeuchi, K.: Virtual reality model of koumokuten generated from measurement. Proceedings Of The Tenth International Conference On Virtual Systems And Multimedia (VSMM'04) (2004) 209–216
10. Yoichi Sato, M.D.W., Ikeuchi, K.: Object shape and reflectance modeling from observation. Procceedings Computer Graphics (SIGGRAPH'97) (1997) 379–388
11. Pulli, K.: Surface Reconstruction and Display from Range and Color Data. PhD thesis, University of Washington (1997)

12. Neugebauer, P.J., Klein, K.: Texturing 3d models of real world objects from multiple unregistered photographic views. Proceedings Euographics '99 **18** (1999)
13. Matsushita, K., Kaneko, T.: Efficient and handy texture mapping on 3d surfaces. Proceedings Euographics '99 **18** (1999)
14. Hendrik P. A. Lensch, W.H., Seidel, H.P.: A silhouette-based algorithm for texture registration and stitching. Graphical Models - IDEAL (2001) 245–262
15. Stamos, I., Allen, P.K.: Geometry and texture recovery of scenes of large scale. Computer Vision and Image Understanding (CVIU) **8** (2002) 94–118
16. Tsai, R.Y.: A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. IEEE Journal of Robotics and Automation **RA-3** (1987) 323–344
17. Hartley, R., Zisserman, A.: Multiple View Geometry in computer vision. CAMBRIDGE (2003)
18. Ilya Eckstein, V.S., Gotsman, C.: Texture mapping with hard constraints. Computer Science Department, Technion Israel Institute of Technology **20** (2001)
19. J.D. Foley, A van Dom, S.F., Hughes, J.: Computer Graphics, Pirnciples ans Practice. 2nd edn. Addison-Wesley Publishing Company (1992)
20. Besl, P.J., McKay, N.D.: A method for registration of 3d shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence **14** (1992) 239256
21. C. Rocchini, P. Cignoni, C.M., R.Scopigno: Multiple textures stitching and blending on 3d objects. Proceedings 10th Eurographics Workshop Rendering (1999)
22. Graham D. Finlayson, S.D.H., Hubel, P.M.: Color by correlation: A simple, unifying framework for color constancy. IEEE Transactions on Pattern Analysis and Machine Intelligence **23** (2001) 12091221
23. Felzenszwalb, P.F., Huttenlocher, D.P.: Distance transforms of sampled functions. Technical report, The University of Chicago, Cornell University (2004)

Segmentation of Triangular Meshes Using Multi-scale Normal Variation

Kyungha Min[1,⋆] and Moon-Ryul Jung[2,⋆⋆]

[1] Sangmyung Univ., Korea
[2] Sogang Univ., Korea

Abstract. In this paper, we present a scheme that segments triangular meshes into several meaningful patches using multi-scale normal variation. In differential geometry, there is a traditional scheme that segments smooth surfaces into several patches such as elliptic, hyperbolic, or parabolic regions, with several curves such as ridge, valley, and parabolic curve between these regions, by means of the principal curvatures of the surface. We present a similar segmentation scheme for triangular meshes. For this purpose, we develop a simple and robust scheme that approximates the principal curvatures on triangular meshes by multi-scale normal variation scheme. Using these approximated principal curvatures and modifying the classical segmentation scheme for triangular meshes, we design a scheme that segments triangular meshes into several meaningful regions. This segmentation scheme is implemented by evaluating a *feature weight* at each vertex, which quantifies the likelihood that each vertex belongs to one of the regions. We test our scheme on several face models and demonstrate its capability by segmenting them into several meaningful regions.

1 Introduction

It is a very important task to define and obtain meaningful regions from a 3D object. In computer vision and computer graphics, triangular meshes, the polygonal approximation of the smooth shape of 3D object, are one of the most widely used method to represent the shape of a 3D object. Therefore, in most applications, the problem of segmenting a 3D object into meaningful regions in computer vision and computer graphics comes down to defining and generating meaningful regions from triangular meshes.

In this paper, we present a scheme that segments triangular meshes into subregions. For this purpose, we also present the approximation scheme that estimates principal curvatures at the vertices of the meshes. Several researchers have been presented the scheme that approximates principal curvatures on triangular meshes [2,4,7]. In this paper we present a robust and efficient scheme

⋆ Corresponding author: minkh@smu.ac.kr. Prof. Kyungha Min, Dept. of Digital Media, Sangmyung Univ., 7 Hongji-dong, Jongro-gu, Seoul, 110-743, Korea.

⋆⋆ This research is accomplished as the result of the research project for culture contents technology development supported by KOCCA.

that approximates the principal curvatures on triangular meshes by extending the normal variation scheme [3,1] into the multi-scale paradigm.

The second step of the scheme is to build a scheme that classifies the subregions of triangular meshes based on the concept of classical differential geometry, which classifies smooth surfaces into several patches such as elliptic convex, elliptic concave, hyperbolic, or parabolic according to the principal curvatures. In this paper, we consider the difference between the triangular meshes and smooth surfaces and modify the classical definitions to handle the special properties of triangular meshes.

The third step of the scheme is to present a scheme that estimates the *feature weight* at each vertex, which represents the likelihood that each vertex belongs to one of the subregions. weight.

The main contribution of this paper is that we present a bridge between the fundamental algorithms [2,4,7] that estimate geometric properties on the triangular meshes and the advanced researches [3,5,6] that extract features from triangular meshes. Since most of the advanced researches, the scheme that determines the likeliness of features on triangular meshes is not clearly explained. In this paper, we give a very detailed explanation about determining the likeliness of features based on differential geometry and the principal curvatures on triangular meshes.

This paper is composed of the following sections. In section 2, we briefly review the related work. We propose the multi-scale normal variation in section 3. In section 4 and 5, we overview the segmentation scheme for smooth surfaces and suggest how to modify the scheme for the domain of triangular meshes. In section 6, we discuss the formula for estimating feature weight, and we present the results in section 7. Finally, we conclude and suggest the direction of future research in section 8.

2 Related Work

Taubin [7] proposed a method to estimate the principal curvatures of a surface at the vertices of a polyhedral mesh by computing in closed form the eigenvalues and eigenvectors of certain 3×3 symmetric matrices defined by integral formulas. This method is targeted to the polygonal surface of a large number of small faces. Meyer et al. [4] proposed a tool to approximate first and second order differential properties on triangular meshes such as normals and curvatures using averaging Voronoi cells and the mixed Finite-Element/Finite-Volume method. They proved the optimality of their scheme under mild smoothness conditions and demonstrated the numerical quality. Lee and Lee [3] exploited normal variations for the estimation of local curvatures of triangular meshes. They computed the feature energy, which is an approximation of local curvature, at a vertex of a mesh. The feature energy of a vertex is defined as the minimum value of the inner products between the normal vector at the vertex and the normal vectors of the faces adjacent to the vertex. Jung and Kim [1] exploited maximal normal variations to find feature vertices on the mesh. The maximum normal variation at a vertex is defined as the value proportional to the maximum angle between the vertex normal and the normals to the faces adjacent to the vertex. It is an approximation to maximal principal curvature. Min et. al [5] have proposed a scheme that evaluates the feature weight of triangular meshes based on hinge angles of edges. At each vertex, the hinge angles of the incident edges are estimated and evaluated to compute a feature weight, which is the likelihood that the vertex belongs to convex or concave region and how much the region around the vertex is curved.

3 Multi-scale Normal Variation

Normal variation is an approximation scheme for the principal curvatures of triangular meshes [3,1]. In this scheme, the angles between the normal of a vertex and the normals of the faces incident to the vertex are estimated and the maximum value is estimated as the maximal principal curvature and the minimum value as the minimal principal curvature. In this paper, we modify this scheme by using the normals on the incident edges instead of on the incident faces. Since the incident edges are on the incident faces, this scheme is not so different from the conventional normal variation scheme. However, this modification allows an easy extension of the normal variation scheme to multi-scale paradigm. The multi-scale paradigm for feature extraction on point cloud data is suggested by Pauly et al. [6]. We apply the multi-scale paradigm to the normal variation and triangular meshes in this paper.

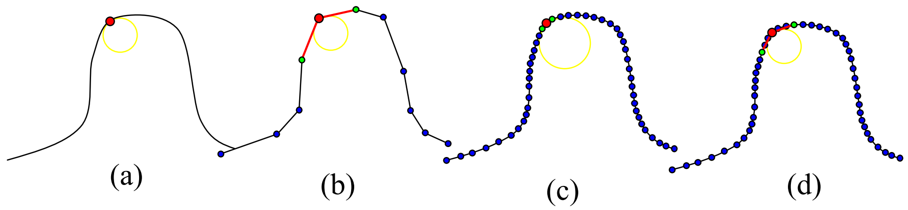


Fig. 1. Multi-scale approximation of the curvatures: (a): the original curve; (b): a coarse approximation of (a); (c) and (d): the same fine approximations of (a). The same point (red point) is sampled on the curves. The radius of the osculating circle (yellow circle), which indicates the curvature at the point, is 1.2 for (a), (b) and (d), and is 1.8 for (c). (b): The curvature at the red point is the same as that of the corresponding point on the original smooth curve. (c): The curvature at the red point estimated using the 1-ring neighborhood is not the same as the corresponding point on the original curve. (d): The same fine approximation where the curvature estimated by using the 3-ring neighborhood is the same as the corresponding point on the original curve.

The multi-scale normal variation is measured by estimating the normal variations of a vertex and its k-ring neighborhood vertices. In case of $k = 1$, the multi-scale normal variation is identical to the classical normal variation scheme. One critical drawback of schemes that estimate principal curvatures at triangular meshes is that the schemes may not able to estimate the principal curvatures for the meshes whose vertices are extremely dense. For the meshes of extremely dense vertex set, the distances between sampled vertices from the smooth surface of 3D object are not enough to represent the principal curvatures of the regions of the sampled vertices (See Figure 1 for an example of this problem). In such a case, the principal curvatures at a vertex should be estimated by considering 2-ring or 3-ring neighborhood vertices.

The result of the multi-scale normal variation is presented in Figure 2. The mesh, which is constructed from the range scanned data of human face, has 73687 vertices and 14305 faces. In Figure 2, the classical normal variation scheme at (a) cannot estimate the principal curvatures at most of the vertices. However, the multi-scale normal variation scheme at (b) and (c) can estimate the principal curvatures at the vertices on the mesh. The difference is the vertices on the nose. Even though the nose is considered a high-curvatured region on a face, the principal curvatures estimated by the conventional normal variation are not so high (Fig 2(b)). The high values of the principal curvatures on the region are clearly estimated by the multi-scale normal variation (Fig 2(c), (d)).

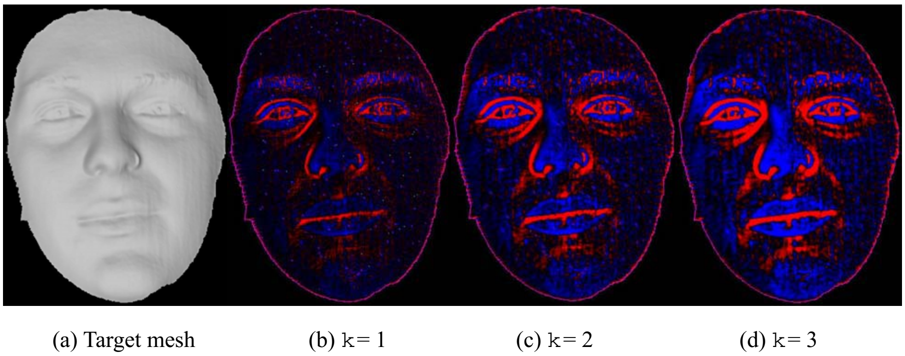


Fig. 2. The result of multi-scale normal variation: (a) The target mesh with 73687 vertices and 146305 faces, (b) principal curvatures estimated by the conventional normal variation where $k = 1$, (c) $k = 2$, (d) $k = 3$

The procedures of estimating principal curvatures in this paper are as follows:

1. Computing normal vectors at each vertex.
2. Determining k for multi-scale normal variation.
3. Estimating principal curvatures using multi-scale normal variation.

3.1 Computing Normal Vectors

At a vertex $\mathbf{v}$ whose incident faces are $\mathbf{f}_0$, $\mathbf{f}_1$, ..., $\mathbf{f}_{N-1}$, the normal vector $\mathbf{n}_\mathbf{v}$ is estimated by the following formula:

$$\mathbf{n}_\mathbf{v} = \frac{1}{N} \sum_{i=0}^{N-1} \frac{\theta_i}{\Theta} \mathbf{n}_i,$$

where $\mathbf{n}_i$ is the normal vector of $\mathbf{f}_i$. θ_i denotes the angle of vertex $\mathbf{v}$ in the triangle $\mathbf{f}_i$ and Θ is the sum of all θ_i's.

3.2 Determining *k*

Determining the value of k, which denotes the level of multi-scaling, depends on the triangular meshes. If the mesh is locally flat (Figure 1 (c)), we need to increase k. The determination of flatness of triangular meshes is estimated by the following strategy. At a vertex of the mesh, we compute k-ring neighborhood of the vertex and estimate the average angles between the normals of the vertex and the vertices on the neighborhood. We repeat this process at each vertex on the mesh and compute the average value from the values on each vertex. If this value is too small, we increase k and repeat this process again until the value is big enough. In this paper, we increase k repeatedly until the average value becomes greater than 0.15.

3.3 Estimating Principal Curvatures

The formula for estimating $k_1(\mathbf{v})$ and $k_2(\mathbf{v})$ at a vertex $\mathbf{v}$ is suggested as follows:

$$\begin{cases} k_1 \leftarrow \max\limits_{1\le i\le n} \{ \ sign(\mathbf{e}_i^{\mathbf{v}}) \ \ |\mathbf{n}_{\mathbf{v}} \cdot \mathbf{e}_i^{\mathbf{v}}| \ \}, \text{for convex} \\ k_2 \leftarrow \min\limits_{1\le i\le n} \{ \ sign(\mathbf{e}_i^{\mathbf{v}}) \ \ |\mathbf{n}_{\mathbf{v}} \cdot \mathbf{e}_i^{\mathbf{v}}| \ \}, \text{for concave,} \end{cases} \tag{1}$$

where $\mathbf{e}_i^{\mathbf{v}}$ is the unit vector from $\mathbf{v}$ to the i-th boundary vertex on the k-ring neighborhood vertices, and $\mathbf{n}_{\mathbf{v}}$ is the normal at $\mathbf{v}$. The $\cdot$ operator denotes the inner product between vectors. The $sign(\mathbf{e}_i^{\mathbf{v}})$ is defined as follows:

$$sign(\mathbf{e}_i^{\mathbf{v}}) \ = \ \begin{cases} +1, \text{if } \mathbf{n}_{\mathbf{v}} \cdot \mathbf{e}_i^{\mathbf{v}} < 0 \\ -1, \text{if } \mathbf{n}_{\mathbf{v}} \cdot \mathbf{e}_i^{\mathbf{v}} > 0. \end{cases} \tag{2}$$

4 Brief Overview on Segmentation of Smooth Surfaces

In differential geometry, a smooth surface can be segmented into the following three patches according to the principal curvatures. In this paper,k_1 denotes the maximal principal curvature and k_2 denotes the minimal principal curvature. Therefore, $k_1 \ \ge \ k_2$.

elliptic: $k_1 k_2 > 0$.
parabolic: $k_1 k_2 = 0$.
hyperbolic: $k_1 k_2 < 0$.

For the elliptic patch, it can be further classified into elliptic convex patch where $k_1 > 0$ and $k_2 > 0$ and into elliptic concave patch where $k_1 < 0$ and $k_2 < 0$. The parabolic patch can be a region such as cylinder where $k_1 > 0$ and $k_2 = 0$. We classify flat region where $k_1 = 0$ and $k_2 = 0$ as parabolic patch. The relationship between the patches is illustrated in Figure 3.

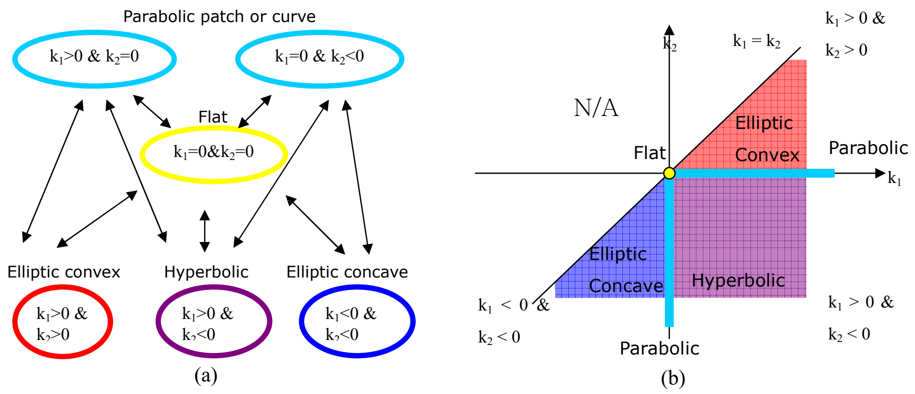


Fig. 3. The relationships between the patches defined in Section 4: (a) The diagram illustrates the definitions and the relationship of the patches. The arrow denotes that the two patches connected by the arrow can be neighborhood, (b) The patches are illustrated in $k_1 - k_2$ plane. Note that the area $\{(k_1, k_2)|k_1 < k_2\}$ is not available (N/A).

5 Segmentation of Triangular Meshes

In this section, we describe four strategies to modify the classification scheme on smooth surfaces presented in Figure 3 into the classification scheme on triangular meshes. Based on the classification scheme, we present a formula that defines the feature weights that will generate meaningful regions from triangular meshes.

5.1 Decomposition of the Hyperbolic

Some of the hyperbolic patches can be more concave than convex ($|k_2| > |k_1|$) and others can be more convex than concave ($|k_1| > |k_2|$). Therefore, we decompose hyperbolic patches into three sub-patches by introducing two parameters t_1 and t_2, where $t_1 > 1$ and $0 < t_2 < 1$.

Convex hyperbolic: $|k_1| > t_1|k_2|$.
Hyperbolic: $t_2|k_2| \leq |k_1| \leq t_1|k_2|$.
Concave hyperbolic: $|k_1| < t_2|k_2|$.

The relation of these three patches is illustrated in Figure 4.

5.2 Modification of Parabolic

According to the modification of the hyperbolic patch, the definition of the parabolic, which is between the elliptic patch and the hyperbolic path should be modified. According to the definition of the border of the two patches, the new definition of the parabolic is as follows:

$$k_1 > 0 \text{ and } k_2 = 0 \longrightarrow k_1 > 0 \text{ and } k_2 < 0 \text{ and } |k_1| = t_2|k_2|.$$
$$k_1 = 0 \text{ and } k_2 < 0 \longrightarrow k_1 > 0 \text{ and } k_2 < 0 \text{ and } |k_1| = t_1|k_2|.$$

The new parabolic on the $k_1 - k_2$ plane are illustrated as cyan lines in Figure 4 (b).

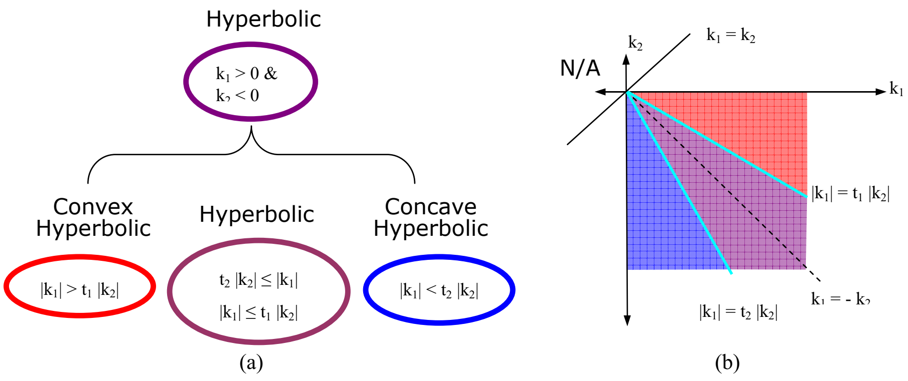


Fig. 4. The decomposition of the hyperbolic patch into three sub-patches: (a) The diagram illustrates how the hyperbolic patch is decomposed, (b) The three sub-patches are illustrated on $k_1 - k_2$ plane

5.3 Modification of Convex and Concave

We define convex patch on triangular meshes by combining the convex elliptic patch and the convex hyperbolic patch. Similarly, the concave patch is defined by combining the concave elliptic patch and the concave hyperbolic patch.

5.4 Modification of Flat

The definition of flat, which is $k_1 = 0$ and $k_2 = 0$, is modified by considering some perturbations of the triangular meshes.

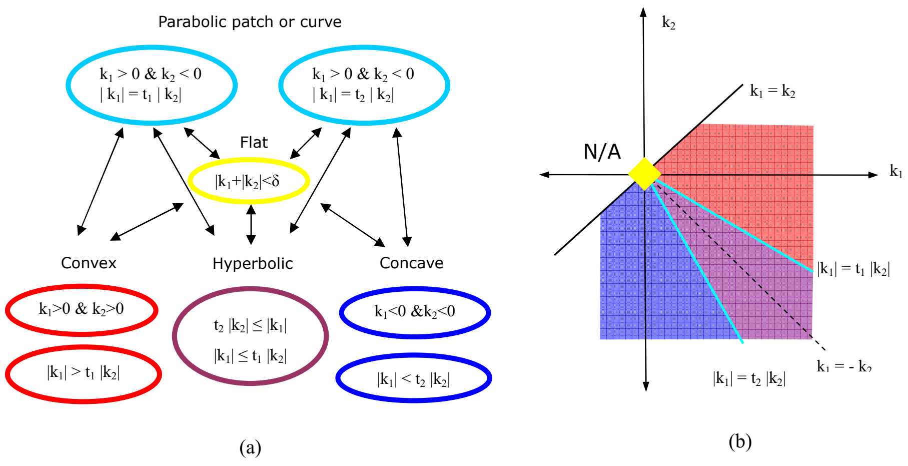


Fig. 5. The diagram of the modified patches: (a) The diagram illustrates the definitions and the relationship of the patches, (b) The patches are illustrated in $k_1 - k_2$ plane

$k_1 = 0$ and $k_2 = 0 \longrightarrow |k_1| + |k_2| < \delta$.

The diagram resulting from the four procedures is illustrated in Figure 5 and Figure 6 briefly illustrates the change of the definitions of the patches in this Section.

6 Estimating Feature Weights

In this section, we build a formula that estimates feature weights on the vertices of triangular meshes. Since the triangular meshes can be segmented according to the scheme presented in Section 5, the feature weight should be defined after considering the segmentation scheme. According to the segmentation scheme in Section 4, k_1 or k_2 can be used as the feature weight. Lee and Lee [3] exploited the minimum normal variation, which is an approximation of k_2 and Jung and Kim [1] exploited the maximum normal variation, which is an approximation of k_1. Therefore, Lee and Lee extracted the extreme concave curves and Jung and Kim extracted the extreme convex curves from triangular meshes. In this paper, our target is to define a feature weight that can define both the convexity and the concavity of the meshes.

Patch	Before modification	After modification
Convex	$k_1 > 0$ and $k_2 > 0$	$k_1 > 0$ and $k_2 > 0$
		$\|k_1\| > t_1\|k_2\|$
Hyperbolic	$k_1 k_2 < 0$	$t_2\|k_2\| \leq \|k_1\| \leq t_1\|k_2\|$
		$\|k_1\| < t_2\|k_2\|$
Concave	$k_1 < 0$ and $k_2 < 0$	$k_1 < 0$ and $k_2 < 0$
Parabolic	$k_1 > 0$ and $k_2 = 0$	$k_1 > 0$ and $k_2 < 0$ and $\|k_1\| = t_2\|k_2\|$
	$k_1 = 0$ and $k_2 < 0$	$k_1 > 0$ and $k_2 < 0$ and $\|k_1\| = t_1\|k_2\|$
Flat	$k_1 = 0$ and $k_2 = 0$	$\|k_1\| + \|k_2\| < \delta$

Fig. 6. The summary of the modification

We present he following strategies for defining the feature weights.

The type of the feature weight: Triangular meshes are classified into three patches: convex, concave, and hyperbolic, which is also known as saddle. Therefore, the feature weight at a vertex should indicate to which patch the vertex belongs.

The magnitude of the feature weight: The magnitude of the feature weight indicates how much the region is curved. We define the magnitude of the feature weight as $\alpha(|k_1| + |k_2|)$, where α is a scaling coefficient.

Consequently, a feature weight at a vertex is defined as a tuple ($t_{\mathbf{v}}$, $u_{\mathbf{v}}$), where $t_{\mathbf{v}}$ is one of $\{convex, concave, saddle\}$, and $u_{\mathbf{v}}$ is a non-negative floating point value. The following formula shows how ($t_{\mathbf{v}}$, $u_{\mathbf{v}}$) is determined.

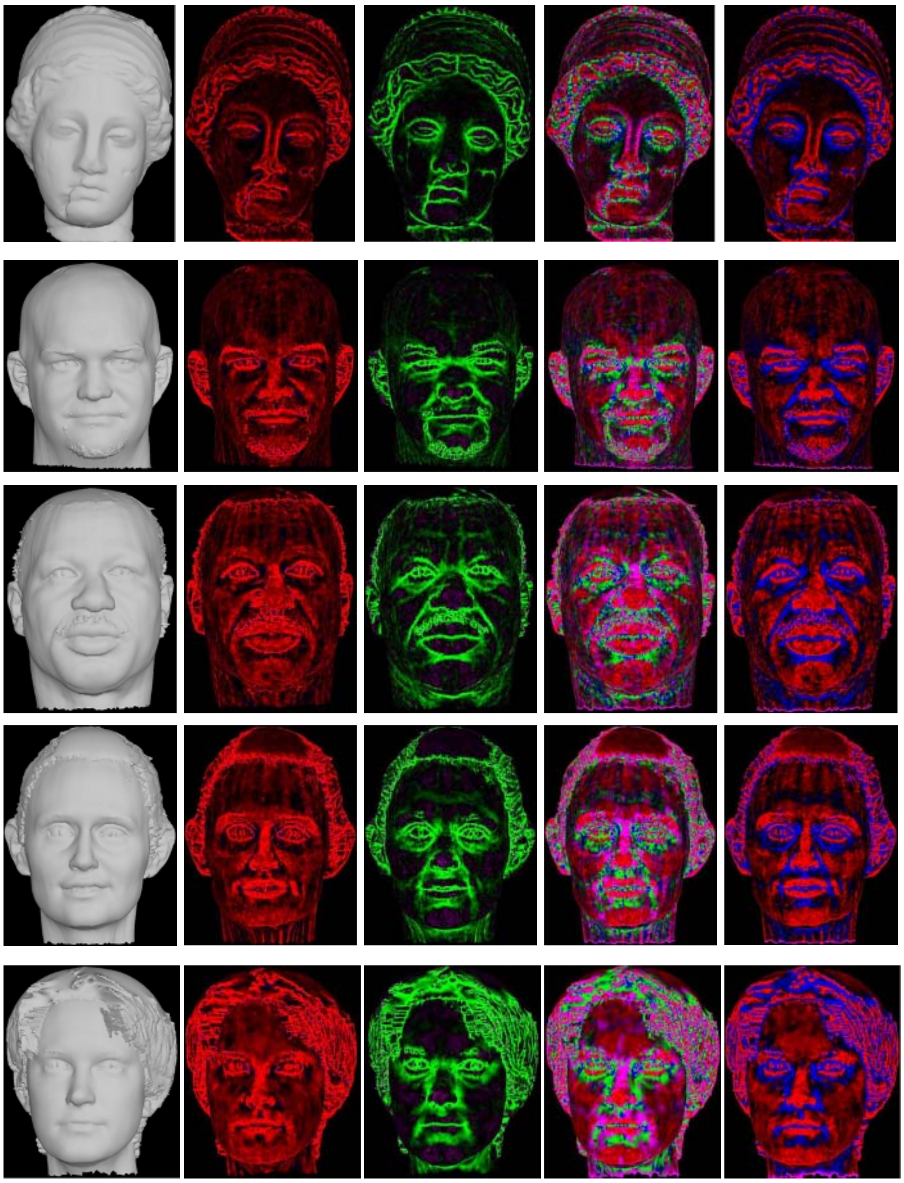


Fig. 7. The result of the algorithm applied to four facial models. The first column is the target model. The second column is rendered by the maximal principal curvatures, and the third one is rendered by the minimal principal curvature. The fourth one is rendered with saddle values. In this figure, red denotes convex region and blue denotes concave. The violet denotes the saddle region where $|k_1| > |k_2|$, while the green denotes the saddle region were $|k_1 < k_2|$. The fifth ones are the targeted segmentation of this paper. The intensity of colors denotes the magnitude of the feature weight.

$$\begin{aligned} t_{\mathbf{v}} &= \begin{cases} convex, & \text{if } \ (k_1 > 0 \text{ and } k_2 > 0) \text{ or } (|k_1| > t_1|k_2|) \\ saddle, & \text{if } \ t_2|k_2| \leq |k_1| \leq t_1|k_2| \\ concave, & \text{if } \ (k_1 < 0 \text{ and } k_2 < 0) \text{ or } (|k_1| < t_2|k_2|), \end{cases} \\ u_{\mathbf{v}} &= \alpha(|k_1| + |k_2|). \end{aligned} \tag{3}$$

7 Implementation and Results

We implemented the proposed algorithm in PC with 3.06 GHz CPU and 2.0MB main memory. We tested the algorithm for five facial models and four general 3D models. The results are illustrated in Figure 7. On every model, the computation of estimating feature weights was executed with in several seconds.

8 Conclusions and Future Work

In this paper, we have presented a novel framework that estimates the geometry of triangular meshes and builds a "segmentation scheme" that segments the meshes into several meaningful subregions. We also improved the robustness of the scheme by extending the normal variation scheme to multi-scale paradigm. But this paper does not report a scheme that automatically extracts regions from triangular meshes, but a scheme that numerically define such regions on triangular meshes. All the examples that show regions are created simply by displaying feature values. As a future work, we will present a feature extraction scheme based on the feature weights estimated in this paper.

References

1. Jung, M. and Kim, H.: Snaking across 3D Meshes. Proceedings of Pacific Graphics 2004. (2004) 415–420.
2. Kobbelt, L.: Discrete Fairing. Proceedings of 7th IMA Conference on the Mathematics of Surfaces. (1997) 101–131.
3. Lee, Y. and Lee, S.: Geometric snakes for triangular meshes. Computer Graphics Forum. **21(3)** (2002) 229–238.
4. Meyer, M. and Desbrun, M. and Schroder, P. and Barr, A.: Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. Proceedings of International Workshop on Visualization and Mathematics. (2002) 35–58.
5. Min, K. and Metaxas, D. and Jung, M.: Active contours with level-set for extracting feature curves from triangular meshes. Proceedings of Computer Graphics International 2006. (2006) 185–196.
6. Pauly, M. and Keiser, R. and Gross, M.: Multi-scale Extraction on Point-sampled Surfaces. Computer Graphics Forum. **22(3)** (2003) 281–289.
7. Taubin, G.: Estimating the Tensor of Curvature of a Surface from a Polyhedral Approximation. Proceedings of International Conference on Computer Vision (ICCV). (1995) 902–907.

Integration of Multiple Methods for Class and Specific Object Recognition

Al Mansur, Md. Altab Hossain, and Yoshinori Kuno

Department of Information and Computer Sciences, Saitama University,
255 Shimo-Okubo, Sakura-ku, Saitama-shi, Saitama 338-8570, Japan
{mansur, hossain, kuno}@cv.ics.saitama-u.ac.jp

Abstract. Service robots need object recognition strategy that can work on various objects and backgrounds. Since no single method can work well in various situations, we need to combine several methods so that the robots can use an appropriate one automatically. In this paper we propose a scheme to classify situations depending on the characteristics of object of interest, background and user demand. We classify the situations into three categories and employ different techniques for each one. We use SIFT and biologically motivated object recognition techniques developed by Serre et al. for two categories. These two methods do not work well on the remaining category of situations. We propose a contour based technique for this remaining category. Through our experiments, we show that the contour based method performs better than the previously mentioned two methods for this category of situations.

1 Introduction

Helper robots or service robots have attracted much attention of researchers for the handicapped or aged people. We are developing a service robot that can find out a specific or a general class of object ordered by the user. For example, if a user asks a robot to find a ‘coke can’, then his/her demand is for a *specific* object and if he/she asks to find any ‘can’, then his/her demand is for a *class* of object. The robots need a vision system that can work on various objects and backgrounds. There is no single object recognition method that can work on various types of objects and backgrounds perceived by such robots. In this paper we present scenarios that have been encountered by a service robot to carry out its object recognition task and propose a solution for these challenges.

It is crucial for the service robot to extract relevant features from the object of interest in order to achieve successful object recognition method. Recently feature extraction by segmentationless methods became much attractive in the object recognition paradigm. Such methods are more robust to complexity of the object and the background. In a recent work [1], Mikolajczyk et al. evaluated the performance of some descriptors that do not require segmentation. They compared these descriptors for robustness against changes of rotation, scale and view point change, image blur, JPEG compression and light change. They concluded that SIFT [2] based descriptors perform best. SIFT is capable of detecting the exact object that the system has previously seen with an incomparable performance. It uses difference of Gaussian detector to find blob-like structures in an object to generate keypoints used for object matching. Unfortunately this method generates very few or no keypoints if the objects are very plain and do not have much detail. Therefore, SIFT is not well suited to recognize such objects. These objects include single color coffee mugs or pieces of fruit in our application domain. SIFT features use positional information in Lowe's algorithm. This algorithm cannot recognize another member of the same class with different texture. For example, to recognize a cup, the system must have seen the same cup or similarly textured cup before. SIFT fails to recognize another cup whose texture is different or even if it does not have any texture. As a consequence, SIFT is not applicable for class recognition. To use SIFT for class recognition, all the position information may be neglected as done in Serre's work [3]. But losing these information, SIFT cannot be considered as our top choice.

Many objects are represented well not by their texture but by their shape. From now on we call them *Simple* objects. These simple objects may be further classified into two groups. First group contains the objects that do not have any texture on them while the second group includes the textured objects although this texture does not characterize them. Objects of the first group will be named as *Simple I* objects and those from the second group will be named as *Simple II* objects.

We name the other objects which cannot be solely described by their shapes as *Complex* objects. For Complex objects we may have many features present, such as texture and patterns on surface and we can use SIFT, Harris corner detector or other similar techniques to extract features. One may say that segmentation is enough for simple objects since simple objects contain no texture. But segmentation is too vulnerable to extract shape information in a realistic background where color of some part of the background is the same as that of the target object.

In a recent work [3], Serre, Wolf and Poggio proposed the standard model that is suitable for class recognition. They claimed that their method is comparable or better than other state-of-the-art techniques and demonstrated that C2 feature is superior to SIFT feature without position information. Although the results are impressive for some object categories, there are some objects for which detection rate is not good enough. Investigation into S1 patches of standard model [3] reveals out that textures in the training or test images produce response from the Gabor filters, which are eventually forwarded to the classifier and the classifier is trained on these textures too. For simple II objects, however, these wrong features train the classifier in a wrong way.

Since the success of an object recognition strategy for a service robot depends on the type of background, type of object and user demand, we have to deploy appropriate technique for a particular situation. We classify situations experienced by a service robot into three categories. The first category consists of the recognition tasks involving specific textured objects from Simple II and Complex objects. SIFT works well for this category. In this paper we use SIFT as described in Lowe's algorithm which includes positional information and is very robust. The second category involves class recognition of Complex objects. Serre's method performs well for this category. Still there is a number of remaining cases which cannot be dealt efficiently with the above mentioned two techniques and these cases have been included in the third category. To deal with the cases of the third category, in this paper, we propose a technique named the contour based method which is an integration of contour detection and

Serre's method. The robot uses SIFT, Serre's method and the contour based method for the first, second and third category respectively. Our proposed classification scheme enables the robot to choose the appropriate detection method. Through experiments, we show that our proposed contour based method performs better than SIFT and Serre's method for the third category of situations. We introduce the classification scheme in section 2 and contour based technique in section 3.

2 Classification of Situations

An object recognition problem can be classified into several categories depending on the nature of background, object and application. Considering the vision system of a service robot as application, we can categorize the recognition problem as in Table 1. We also mentioned the applicability of SIFT, Serre's method and the contour based method for each case. We choose SIFT and Serre's method since these are two of the state-of-the-art object recognition methods and each is dedicated to different scenario.

We define the terms used to represent background types and object types in Table 1 as follows:

Background type:
Simple: single color background; segmentation can easily remove the background.
Complex: multicolor, textured or patterned background
Object type:
Simple I: single color, textureless object. Example: single color fruit, ball, etc.
Simple II: some members may have texture while others do not. Example: mug, cup etc.
Complex: textured or patterned and this texture or pattern is required for their recognition. Example: keyboard, piano, etc.
Object specificity:
Specific: particular instance of an object
Class: any member of an object category

As discussed in the introduction, SIFT is capable of efficiently recognizing a specific Complex or textured Simple II object. It is not suitable for Simple I, Simple II (textureless) objects or class recognition.

Examining the S1 patches (Fig. 1) of Serre's method, we notice that many strong responses come from the textures on the target object and also from the background. These responses are ultimately used for training, and the classifier considers them as a part of a positive example. As a result Serre's method cannot be used effectively to recognize Simple I or Simple II (textureless) objects in complex background and Simple II (textured) objects in any background.

In this paper we propose a way to deal with cases 1, 2(b), 3, 6, 7 (b) and 8. Since the shape is the fundamental characteristic of Simple I and Simple II objects, we embed contour detection in Serre's method to eliminate unwanted textures. Although the contour detector is not able to remove all the texture completely, the recognition rate improves considerably. To detect the contour we use nonclassical receptive field inhibition [4].

Table 1. Categorization of an object recognition scenario depending on the type of background, object and user demand

Backgro-und type	Object type	Specific/ class	Case	Applicability		
				SIFT	Serre's method	Contour based method
Simple	Simple I	specific	1			•
		class				
	Simple II	specific (textured)	2 (a)	•		
		specific (textureless)	2 (b)			•
		class	3			•
	Complex	specific	4	•		
		class	5		•	
Complex	Simple I	specific	6			•
		class				
	Simple II	specific (textured)	7 (a)	•		
		specific (textureless)	7 (b)			•
		class	8			•
	Complex	specific	9	•		
		class	10		•	

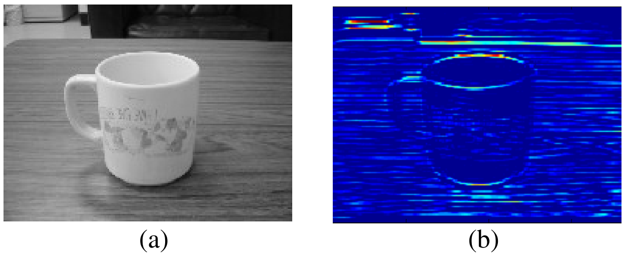
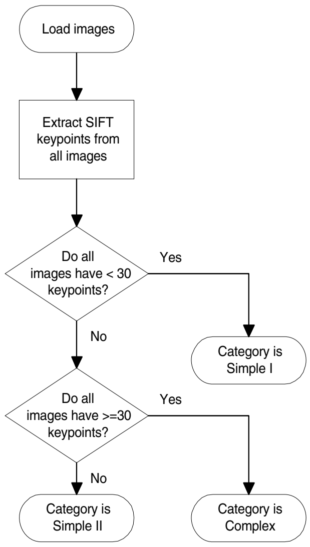


Fig. 1. (a) Input image (b) 0 degree S1 patch [3] showing the response generated by textures on background and cup surface

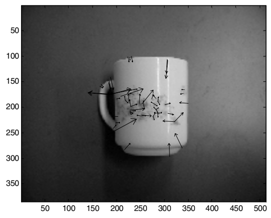
To categorize scenarios into one of the 12 cases, we need three kinds of information: background type, object type and object specificity. We apply the algorithm shown in Fig. 2(a) to classify an object into Simple I, Simple II or Complex categories. In this algorithm, we use diverse samples where the targets exist in a simple background to ensure that no keypoint is generated from the background.

When the user asks the robot to find an object, the robot remembers which category that object falls into. If the demand is specific and the object can be classified as Simple II, robot remembers whether the object had texture or not. The robot is trained on all the objects (on which the robot works) using the algorithm shown in Fig. 2 (a) prior to recognition. Since any suitable method for complex background can also work in simple background, we can use the same algorithm for both types of backgrounds if the object type and object specificity remain the same and we need not to classify the background. Finally, object specificity will be known from the robot user. Now we deploy appropriate strategies for three categories of cases as follows:

SIFT: cases 2 (a), 4, 7 (a), and 9 (Category 1)
Serre's method: cases 5 and 10 (Category 2)
Contour based method: cases 1, 2(b), 3, 6, 7 (b) and 8 (Category 3)


Load images
Extract SIFT keypoints from all images
Do all images have < 30 keypoints?
Yes
No
Category is Simple I
Do all images have >=30 keypoints?
Yes
No
Category is Simple II
Category is Complex

(a)


(b)

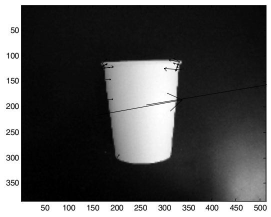

(c)

Fig. 2. (a) Object classification algorithm (b) One example of a cup have 67 keypoints (c) another example have 17 keypoints

We show the classification of 'cup' as an example (Figs. 2(b) and (c)). In this example, one sample from 'cup' category has 67 keypoints and another has 17. As a result, not all cups have greater than or equal to 30 keypoints. Consequently it is classified as a Simple II object. The threshold of 30 keypoints has been found by extensive experiments on various types of object categories.

3 Contour Based Method

Our contour based method consists of two major steps: (1) Contour detection and (2) Feature extraction, training and classification.

3.1 Contour Detection

At the first step we extract shape information by eliminating textures from the object and background. For this purpose, we use biologically inspired contour detection technique proposed in [4], which is called nonclassical receptive field (non-CRF) inhibition. This method offers improved contour detection in our robot vision. There is evidence that 80% of the orientation-selective neurons in the primary visual cortex of monkeys exhibited non-CRF inhibition. We are motivated to use non-CRF inhibition as it seems to be a general property of biological edge detectors involved in human and other biological perception of edges and lines. This mechanism extracts the edges that belong to the contours of objects while suppressing edges which belong to texture regions. Two types of inhibitory mechanism have been proposed in [4]: isotropic and anisotropic inhibition. We use isotropic inhibition to get responses only from isolated lines and edges. This inhibition operator does not respond to lines or edges that are surrounded by textures.

3.2 Feature Extraction, Training and Classification

We employ the feature extraction, training and classification technique proposed in [3], which follows the standard model of object recognition in primate cortex [6]. The standard model in its simplest version is described in [5]. This model can be split into four steps involving computation of four types of responses namely S1, C1, S2 and C2 as given below:

S1: We apply a battery of Gabor filters to the contour extracted images. Filters with 4 orientations and 16 scales have been used. We obtain 16X4 = 64 S1 maps that are arranged in 8 bands. Number of filter orientations can be increased to capture the oriented features more accurately but this will increase the computation time.
C1: Then we take the maximum over scales and positions for each band. These steps ensure scale and position invariance. To obtain the C1 responses, S1 maps are subsampled using a particular grid size for each band. We get one measurement from each grid cell by taking the maximum of all 64 elements. C1 contains four orientation maps. During training only, we extract K patches of various sizes and four orientations from the C1 maps derived from training images.

S2: Then we compute $Y = \exp(-\gamma \| X - P_i \|^2)$ for image patches X and training patches P_i. By doing this we get S2 maps.
C2: Finally we compute the C2 responses by taking a global maximum over all scales and positions for each S2 type at each position on the S2 map.

We train a Support Vector classifier using these C2 features and the trained classifier is used for recognition. The outcome of this whole sequence of steps is a trained recognizer for contoured objects.

4 Experimental Results

We evaluated our object detection technique by using four object categories: chair, cup, ewer and ibis. These datasets are from Caltech database which are available at www.vision.caltech.edu. These four categories are included in the ten worst case categories in [3]. We carried out the experiments as follows: each of the four datasets was split randomly into two sets: training set and test set. Each set contained 30 images. First set was used for training and the second one was for testing. The negative training and test sets were randomly generated from the background images as in [3]. These background images were collected from junk images from the internet which were totally unrelated to the keyword category. Each experiment was carried out under identical conditions. For some datasets, less than 30 images were left after training images were drawn. In this case, we took mirror images of some images to make the number of images 30. For each category, we repeated the experiments several times for each value of feature (using randomly chosen 30 training images each time) to obtain an unbiased estimate of performance. We used contour detection software available at http://matlabserver.cs.rug.nl and the code for biologically motivated object recognition system available at http://cbcl.mit.edu. For contour detection, we used the following values for different parameters:

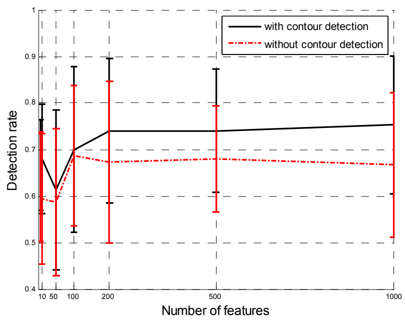
Wavelength: 4; Initial orientation: 0^o; Phase offset: 0^o, 90^o; Aspect ratio: 0.5; Bandwidth: 1; Number of orientations: 16; Inhibition type: isotropic surround inhibition; Superposition for isotropic inhibition: L-infinity norm; Alpha: 2; DoG parameters: $K_1 = 1$, $K_2 = 4$; Thinning: enabled.

To classify the studied objects into Simple I, Simple II or Complex categories we used single color background images so that we could use our classification algorithm. We used our own images if there were no such images in the dataset. Chair, cup and ewer were classified as Simple II objects. For ibis, we got greater than 30 keypoints from all of the images. But sometimes the number of keypoints was close to 30. So we tried the contour based method to recognize it although it had been classified as a Complex object. In the class recognition results given in Table 2 we demonstrate the superiority of contour based method over Serre's method for the three object categories where the situations falls into case 8 as mentioned earlier. Since SIFT is not suitable for class recognition, we omit its performance in this table.

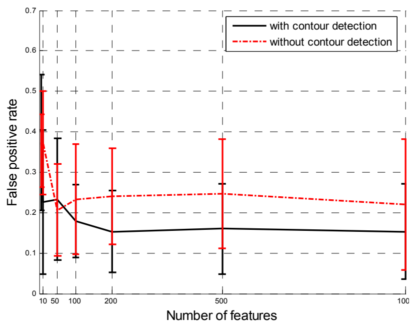
In Fig. 3 we compared the performance of the contour based and Serre's method for 'cup' category. We used identical parameters to make the comparison fair. The numbers of training and testing samples were kept the same throughout. We found that the modified system outperforms the original system for the studied cases.

Table 2. Performance comparison of Serre's and the contour based method

Object category	Serre's method (no. of features=250)		Contour based method (no. of features=250)	
	Detection rate, %	False positive rate, %	Detection rate, %	False positive rate, %
Chair	62	24	76	14
Cup	67.3	24	74	15.3
Ewer	74.7	24	80.7	14.7
Ibis	66	29.3	63.3	16.7



(a)



(b)

Fig. 3. Performance comparison: (a) Detection rate (b) False positive rate

Fig. 4. Examples of test images used in experiments

Performance results for other objects where both systems performed equally are not shown. These results verify that use of contour detection for Simple I and Simple II objects can significantly improve the detection rate and lower the false positive rate. Time required for contour detection is negligible compared to the time required in later stages. As a result it does not add any significant burden to the system.

5 Conclusion

To make a service robot's vision system work well in various situations, we have integrated several methods so that robot can use the appropriate one. We have proposed a scheme to classify the situations depending on the characteristics of object of interest, background and user demand. It has been shown that it is possible to classify the situations into three categories and employ separate techniques for each group. This classification scheme enables a service robot to automatically decide the appropriate detection method to use. SIFT and Serre's method have been employed for two categories. A contour based technique for the third category has also been proposed. We verified that the contour based method performs better than the previously mentioned two methods for this category of situation.

In some object category, all members may be textured although these textures vary much within samples and they are mainly described by their shapes. In this case categorization into Simple II or Complex object is difficult using SIFT keypoint count. We are investigating the use of PCA to detect the similarity of textures and to solve this problem.

Acknowledgement

This work was supported in part by the Ministry of Internal Affairs and Communications under the Strategic Information and Communications R&D Promotion Program and by the Ministry of Education, Culture, Sports, Science and Technology under the Grant-in-Aid for Scientific Research (KAKENHI 18049010).

References

1. Mikolajczyk, K., Schmid, C.: A Performance Evaluation of Local Descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27 (10) (2005) 1615 – 1630.
2. Lowe, D.: Distinctive Image Features from Scale-invariant Keypoints. International Journal of Computer Vision, 60(2) (2004) 91-110.
3. Serre, T., Wolf, L. and Poggio T.: A new biologically motivated framework for robust object recognition. Ai memo 2004-026, cbcl memo 243, MIT, 2004.
4. Grigorescu, C., Petkov, N. and Westenberg, M. A.: Contour detection based on nonclassical receptive field inhibition. IEEE Trans. on Image Processing, 12 (7) (2003) 729-739.
5. Riesenhuber, M., and Poggio, T.: Hierarchical models of object recognition in cortex. Nature Neuroscience, 2(11) (1999) 1019–25.
6. Riesenhuber, M., and Poggio, T.: How visual cortex recognizes objects: The tale of the standard model. The Visual Neurosciences, 2 (2003) 1640–1653.

An Efficient Photon Mapping Algorithm for Rendering Light-Emitting Fluids

Kyungha Min*

Dept. of Digital Media, Sangmyung Univ.,
7 Hongji-dong, Jongro-gu, Seoul, 110-743, Korea
minkh@smu.ac.kr

Abstract. In this paper, we present a novel and efficient algorithm for rendering light-emitting fluids such as fire. For this purpose, we extend the well-known photon mapping algorithm for volumetric environments. The photons emitted from the fluids are stored in a voxelized space, instead of k-d tree in the original photon mapping scheme. We further propose an efficient photon collecting algorithm based on Bresenham's algorithm, which can collect photons in reduced computational loads. We prove the effectiveness and efficiency of the proposed algorithm by visualizing light-emitting fluids such as fire in various styles.

1 Introduction

Light-emitting fluids such as fire, hot smoke, and explosions are one of the most interesting topics in computer graphics. The animation techniques for such fluids require a rendering scheme as well as a simulation scheme. Building fluid animations based on fluid equations such as Navier-Stokes equations [6], semi-Lagrangian scheme [13] and Euler equations [4] have been proposed. Furthermore several researchers have been proposed the physically-based simulation schemes for the light-emitting fluids such as fire [2, 12, 11, 10] and explosion [5].

The most widely-used algorithm for visualizing fluids is the photon mapping scheme. The photon mapping scheme casts photons from light source to the environment and stores the photons on the surfaces in a data structure such as k-d tree [7]. In the rendering step, Jensen presented a spherical scheme for sampling photons [7]. Several points on the ray from the view point are sampled and photons that lie inside a *sampling distance* from the point are collected to make contributions in computing the color of the pixel which corresponds to the ray. This method, however, has the problem that the photons can be sampled irregularly and the inefficiency that the sampling process requires distance computations between the point on the ray and the photons. Since millions of photons are cast in many cases, this distance computation becomes one of the major computational load in the photon mapping rendering scheme.

We extend the photon mapping scheme in the following two points. First, **we store and manage photons in a voxelized environment instead of k-d**

* Corresponding author.

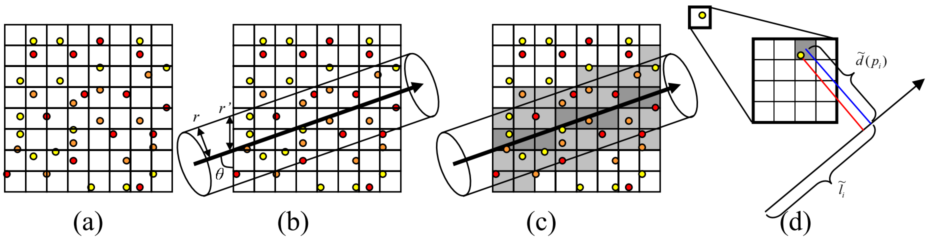


Fig. 1. Basic ideas of this paper: (a) Photons are stored in a voxelized environments, (b) Photons near a ray are sampled in a cylindrical sampling scheme, where r denotes the sampling distance, (c) The voxels within r from a ray are determined by exploiting Bresenham's algorithm, (d) A voxel is decomposed into several sub-voxels and the photon is stored as the index of the sub-voxel

tree (See Figure 1 (a)). Since most of the fluid simulations are implemented in a voxelized environment, it is natural to store photons that will visualize the fluids in the same environment. Furthermore, instead of storing the exact position of the photon, we present an efficient scheme that decomposes the voxels into several sub-voxels and stores the index of the sub-voxel where the photon is located (See Figure 1 (d)). This scheme reduces memory storage for storing photons and improves the computation for sampling photons. Second, **we sample photons near the ray from the viewpoint in a cylindrical sampling scheme instead of the spherical sampling scheme** (See Figure 1 (b)). Adabala and Manohar also presented similar sampling scheme [1].

The most important benefit of the cylindrical scheme is that it guarantees uniform sampling of the photons than the spherical scheme. Additionally, we achieve another benefit by combining the scheme with Bresenham's algorithm, which is the well-known line rasterization algorithm in 2D. By modifying and applying the Bresenham's algorithm to the voxelized space and the ray, we can determine the voxels that the ray pass through efficiently. Furthermore, an extension of this scheme allows those voxels that lie within the sampling distance to be computed without computing their distances from the ray (See Figure 1 (c)). Finally, we introduce a scheme that approximates the distance between a photon and the ray by the distance between the sub-voxel and the ray (See Figure 1 (d)). This scheme reduces the computational loads in a great scale without decreasing the rendering quality. Another benefit of this scheme is that we can achieve a constant rendering time even though we increase the number of photons. We apply the proposed rendering algorithm to the fire animation created in [11] and create various rendering effects.

This paper is organized as follows. In Section 2, the schemes for rendering light-emitting fluids and simulating light-emitting fluids are surveyed briefly. In Section 3, we briefly overview the photon mapping algorithm, and we present the rendering algorithm for light-emitting fluids in Section 4. In Section 5, we suggest the implementation details and results. Finally, we conclude this paper and suggest the future works in Section 6.

2 Related Work

2.1 Rendering Light-Emitting Fluids

The original photon mapping algorithm was developed for global illuminations [7], and it was extended to visualize fluids and participating media [8]. The scheme was also further extended to render smoke [4] and fire [9, 12].

2.2 Simulating Light-Emitting Fluids

Most of the animation schemes for fluids are developed based on fluid equations such as Navier-Stokes equations and Euler equation. Among them, the light-emitting fluid such as fire or explosion requires additional schemes such as level-sets [12], combined particle system [5] or simulation of combustion process [11]. In [12], the core of flame is simulated as level-set surface, which is animated according to the projection of the fuel from the source, and the flying plums of fire are represented as the fluids projected from the surface of the level-set surface. In [5], the fuels are modeled as particle systems, and they cooperate with the classical fluid equations for the simulation of explosion and combustion. In [11], the combustion process is simulated in the voxelized environments where fluid equations are solved numerically. Initially the the fluids are assumed to be fuel whose amount is identical to the density. During the simulation, the fuels are combusted and generate heat, which increases the temperature of the fluids. After all the fuels are combusted, the fluids become soot.

3 Overview on Photon Mapping

In this section, we briefly describe the basic mechanism of photon mapping algorithm [8] and explain how the algorithm is modified to render light-emitting fluids in this paper. The photon mapping algorithm renders fluids in the following procedures:

1. **Generating photons.** The light sources generate and cast photons to the environment to render. In the original algorithm [8], only light sources generate and cast photons. In light-emitting fluids, the fluids also play the role of light source. Therefore, the voxels inside the light-emitting fluids generate and cast photons.
2. **Casting photons.** Photons from the light sources are cast uniformly. However, the photons from the light-emitting fluids are cast to the outward directions. The casting direction is determined as follows: First, a random direction is selected. Second, we sample a point in the selected direction whose distance is the length of a voxel. Third, if the temperature of the sampled point is greater than the voxel where the photon is generated, then the direction is discarded and re-select another direction and repeat the above process again until the direction passes the test.

3. **Travelling photons.** During the travels of photons, it has three selections: transit, reflection and absorption. We exploit the Russian Roulette algorithm in [8] for the selection. At each point of selection, the power of the photon is reduced according to the travelled distance and the density of the point. In case that the selection is transit or reflection, the photon is stored at the voxel where the selection is made.
4. **Storing photons.** In this paper, we classify photons according to the place where the photon is stored. The photon stored at the voxel where it is generated is denoted as a *fluid photon*, and the photon stored in the voxel which is not occupied by the fluid is denoted as an *air photon*. Finally, the photon stored on the surface of an object is denoted as a *surface photon*. In [8], the positions of the photons are managed by k-d tree. However, we manage the positions in the voxelized spaces, since the space is already voxelized for the purpose of the numerical solution of the fluid equations.
5. **Efficiency-improving scheme.** Most of the computational loads in rendering step come from the process of sampling photons, which includes the computation of the distances between photons and the ray. In this paper, we present an approximation scheme for estimating the distance. For this purpose, we decompose each voxel into eight sub-voxels (See Figure 1 (d)). For a better precision, the voxel can be decomposed into sixty four sub-voxels, which is one step further decomposition from the eight sub-voxels. Instead of storing the position of the photons, we store the index of the sub-voxel that contain the photon. The index has the value of (a, b, c), where the value is $\{0, 1\}$ for eight sub-voxels and $\{0, 1, 2, 3\}$ for sixty four sub-voxels. Therefore, the distance between the photon and the ray is approximated by the distance between the sub-voxel and the ray (In Figure 1 (d), blue line replaces red line).

4 Rendering

The rendering scheme that visualizes light-emitting fluids is developed based on the ray marching scheme for volume rendering [8, 12]. As a preliminary step, we build a set of temperature-color graphs that determine the colors of the light-emitting fluid based on the temperature of the fluids.

4.1 Determination of Color

In related researches [9, 12], the color of light-emitting fluids such as fire is determined based on the black body radiation. Even though their scheme provides a physically correct way to determine the colors of the fluids, they do not give the control of determining colors to the fluids. In many application fields such as movies, animations and games, animators may want to create the color of the fluids in their own style. Therefore, we present a temperature-color graph which is a controllable approach in determining the color of light-emitting fluids. In designing the temperature-color graph, we exploit the field function defined for building soft objects [3]:

$$f(d) = \begin{cases} 1 - \dfrac{(3d^2)^2}{p + (4.5 - 4p)d^2}, & 0 < d \leq \dfrac{1}{2} \\ \dfrac{(1 - d^2)^2}{0.75 - p + (1.5 + 4p)d^2}, & \dfrac{1}{2} < d \leq 1 \\ 0, & 1 < d \end{cases}$$

One benefit of the field function is that we can control the slope of the curve through the parameter p.

In mapping temperature to color, we set three threshold temperatures: τ_0, τ_1, and τ_2, for each color. Each of the threshold temperatures are matched for $d = 0$, $d = 0.5$, and $d = 1$ of the field function.

$$\mathbf{C}(T) = \begin{cases} 0, & \text{if } T < \tau_0 \\ f(d_1), & \text{if } \tau_0 \leq T < \tau_1 \\ f(d_2), & \text{if } \tau_1 \leq T < \tau_2 \\ 1, & \text{otherwise,} \end{cases}$$

In the above formula, d_1 and d_2 are defined as:

$$\begin{aligned} d_1 &= 1 - \frac{1}{2}\frac{T - \tau_0}{\tau_1 - \tau_0} \\ d_2 &= \frac{1}{2}\left(\frac{T - \tau_1}{\tau_2 - \tau_1} + 1\right). \end{aligned}$$

An example of the temperature-color graph is illustrated in Figure 2. Three temperature-color graphs ($C_R(T)$, $C_G(T)$, $C_B(T)$) for three colors (Red, Green, Black) are illustrated. At the bottom, the resulting color band from the graphs is illustrated. Users can edit the curve by modifying τ_0, τ_1, and τ_2 values for each color or p value for each function. By editing the graph, users can control various rendering effects of fire.

4.2 Integration

Determining voxels. The first step of integration is to determine the voxels that contain the photons near the ray. The procedures of the determination is executed in the following steps:

1. As a preparation, we assign the sampling distance, which is denoted as r in Figure 1 (b), and the viewing information such as view point and viewing direction. We assume that the view point lies outside the voxels that enclose the fluids to visualize. From this assumption, the *first plane*, which is defined

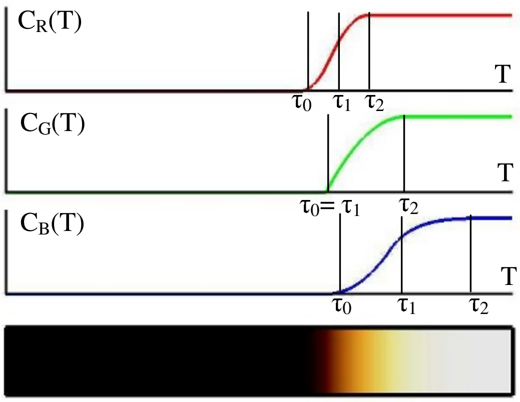


Fig. 2. Temperature-Color graph

as the plane of the voxels that intersect with ray for the first time, is one of the six planes $\{x = 0,\ x = N_x - 1,\ y = 0,\ y = N_y - 1,\ z = 0,\ z = N_z - 1\}$, where $N_x,\ N_y,\ N_z$ are the size of the voxels.

2. At the first plane, we compute the *first voxel* that intersects the ray. Among the voxels on the first plane, we select the voxels that lie inside the modified sampling distance r' from the first voxel. Note that r' is computed as $r' = r\cos\theta$ and θ is the angle between the viewing direction and the first plane (See Figure 1 (b)). Those voxels on the first plane that lie within r' are exploited in determining the voxels within the modified sampling distance in the next planes.
3. At the next plane, we apply Bresenham's algorithm to determine the voxel that lie on the ray. Then, we compute the displacement of the voxels on the ray between the previous plane and this plane, and apply it to the voxels within the modified sampling distance of the previous plane and determine the voxels within the sampling distance on next plane. We repeat this process to the *last plane* and determine all the voxels that lie within the sampling distance.

This process is visualized in Figure 3.

Sampling photons. After we have determined the voxels inside the sampling distance, we collect the photons that lie inside the sampling distance and estimate their distances from the ray. Before computing the distance between the photons and the ray, we project the centroid of the sub-voxel onto the ray and compute $\tilde{d}(p_i)$, the distance between the centroid of the ray, which substitutes $d(p_i)$ in the Equation (1), where p_i is a photon, and $\tilde{l}$, the distance between the projected centroid and the view point, which substitutes l_i in the Equation (1) (See Figure 1 (d)). For each photon, we search the voxel and its sub-voxel that contains the photon, and exploit the pre-computed values for the distance from the ray and the distance from the view point.

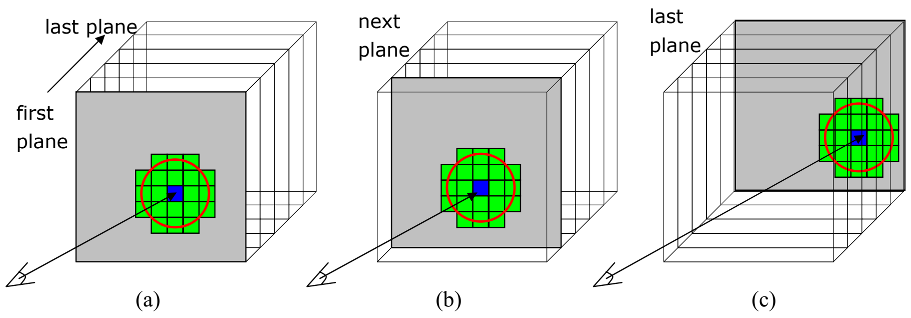


Fig. 3. Determining voxels inside the sampling distance: (a) the first voxel (blue rectangle) and its surrounding voxels (green rectangles) that lie inside the modified sampling radius $\mathbf{r}'$ (red circle) are computed; (b) the sampling voxels in the next plane are computed using the Bresenham's algorithm; (c) this process is repeated until the ray intersects the last plane

Integration. The color of a pixel is determined by integrating the colors of the photons lying inside the cylinder whose axis is the viewing direction and radius is the sampling distance r. The integration equation is illustrated as follows:

$$\mathbf{I} \;=\; \frac{1}{n}\sum_{i=1}^{n} e^{-k_1 l_i^2} d(p_i)^{-k_2}\mathbf{C}(p_i), \tag{1}$$

where $\mathbf{I}$ denotes the color vector at a pixel, and n denotes the number of sampled photons. $d(p_i)$ denotes the distance of the i-th photon from the viewing direction and l_i denotes the distance from the view point. $\mathbf{C}(p_i)$ denotes the color of the i-th photon, which comes from the temperature-color graph. k_1 and k_2 are parameters that control the effects of the distance from the view point and the distance from the viewing direction on the color, respectively.

4.3 Lighting Effects

The lighting effect of fire is implemented by applying the color of photons stored on the surface of objects during integration process. Therefore, the integration equation presented in Equation (1) is modified as follows:

$$\mathbf{I} \;=\; \frac{1}{n}\sum_{i=1}^{n} e^{-k_1 l_i^2} d(p_i)^{-k_2}\mathbf{C}(p_i) \;+\; e^{-k_3 l(q)^2}\mathbf{C}(q), \tag{2}$$

where q is the intersection point between the viewing direction and the object, and $\mathbf{C}(q)$ is the color at q and $l(q)$ is the distance between the view point and q. Note that the decreasing factor k_3 depends on the status of the intersection voxels from the view point to q. k_3 depends on the densities of the intersecting voxels. $\mathbf{C}(q)$, the color of q, is estimated in the following formula:

$$\mathbf{C}(q) \;=\; \mathbf{C}_q * (\mathbf{P}(q) + \mathbf{A}), \tag{3}$$

where $\mathbf{C}_q$ is the material color of the object at q and $\mathbf{P}(q)$ is the color of the surface photons that lie within the sampling distance. $\mathbf{A}$ is the color of the ambient light. The multiplication $\mathbf{V} * \mathbf{W}$ denotes the component-wise multiplication, which means $(v_1 * w_1, ..., v_n * w_n)$, where $\mathbf{V} = (v_1, ..., v_n)$ and $\mathbf{W} = (w_1, ..., w_n)$.

5 Implementation and Results

The proposed algorithm in this paper is implemented at a Pentium-based PC with 3.06 GHz CPU and 2.0 GByte Main memory. The light-emitting fluids visualized in this paper is a fluid that represents fire, which was developed in [11]. We compare the three different rendering schemes: (i) the original photon mapping algorithm that exploits k-d tree for photon storing and spherical sampling scheme, (ii) a photon mapping algorithm that exploits the voxelized structure for photon storing and cylindrical sampling scheme but not the sub-voxel approach, and (iii) a photon mapping algorithm that exploits the voxelized structure, cylindrical sampling scheme and the sub-voxel approach. We test the presented algorithm to create two different rendering results: (i) visualizing fire and (ii) visualizing fire with environment.

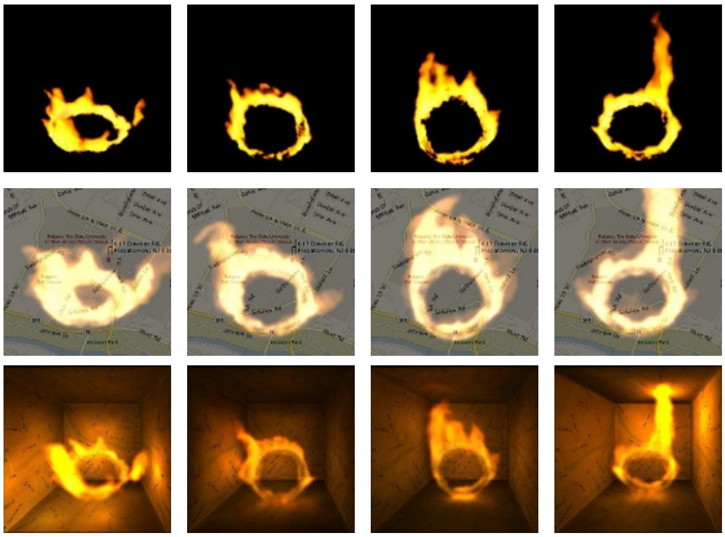
Figure 4 illustrates the rendering results of the proposed algorithm. Figure 4 (a) is the rendering result of fire from rotating torus. The images in the first row is rendered by the conventional volume visualization scheme [11]. The images in the second row are rendered by considering the background image. The images in the third row is rendered by simulating the lighting effect on the boundaries suggested in Section 4.3. Figure 4 (b) is the rendering result of fire from a fixed source. The images in the first row is rendered by the conventional volume visualization scheme [11]. The images in the second row are rendered by considering the lighting effect to the air, where the photons cast from fire are stored in the air. The images in the third row is rendered by simulating the lighting effect on the boundaries suggested in Section 4.3.

The rendering styles are implemented in the following strategies:

Rendering with background: The color of a pixel is determined by blending the integrated color of the photons and the color of the background according to the well-known alpha blending scheme.

Lighting effect to the air: The photons cast from fire are also stored in the voxels that does not contain any fluids. In this case, the range of lighting effect is controlled by the decreasing of the photons in the air. At the integration step, those photons stored in the air play the role of lighting effects to the air.

Lighting effect to the environment: The photons cast from fire are stored on the surfaces of the background. At the integration step, those photons are integrated according to the Equation (3). For the lighting effect of some ambient light, we precast the photons corresponds to the ambient light before casting photons from light-emitting fluids.


(a) Rendering fire from rotating torus in various styles

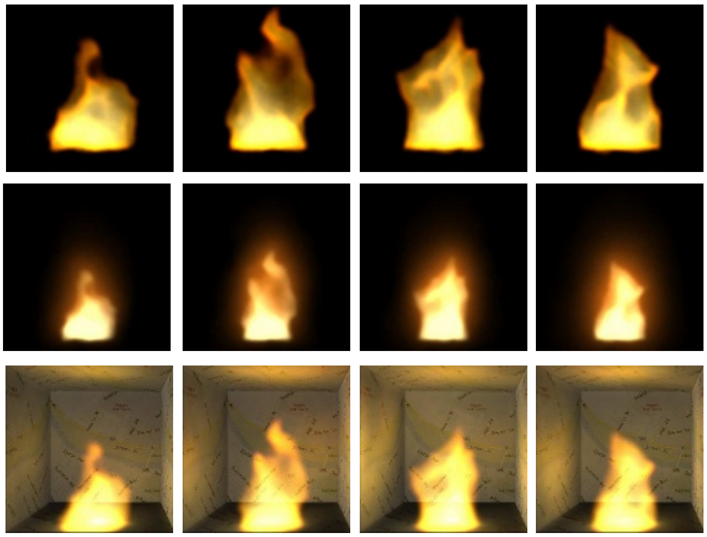

(b) Rendering fire from fixed source in various styles

Fig. 4. Three styles of rendering results for two types of fires

6 Conclusions and Future Work

In this paper, we have presented an efficient photon mapping algorithm for rendering light-emitting fluids and have proved the efficiency and excellence of the algorithm by rendering fire developed in [11].

In the future work, we are going to extend this rendering scheme to visualize several heterogeneous fluids simultaneously and to develop a rendering scheme for a complex scheme that includes shiny objects and transparent objects.

References

[1] Adabala, N. and Manohar, S.: Modeling and rendering of gaseous phenomena using particle maps. Journal of Vis. and Comp. Anim. **11(5)** (2000) 279–293.

[2] Adabala, N. and Hughes, C.: A Parametric model for real-time flickering fire. Proceedings of Computer Animation and Social Agents (CASA 2004). (2004).

[3] Blanc, C. and Schlic, C.: Extended field functions for soft objects. Proceedings of Implicit Surfaces 1995. (1995) 21–35.

[4] Fedkiw, R. and Stam, J. and Jensen, H.: Visual simulation of smoke. Proceedings of SIGGRAPH 2001. (2001) 15–22.

[5] Feldman, B. and O'Brien, J. and Arikan, O.: Animating suspended particle explosions. ACM Transactions on Graphics. **22(3)** (2003) 708–715.

[6] Foster, N. and Metaxas, D.: Modeling the motion of a hot, turbulent gas. Proceedings of SIGGRAPH 1997. (1997) 181–188.

[7] Jensen, H.: Global illumination using photon maps. Proceedings of Rendering Techniques '96 (Proceedings of the 7th Eurographics Workshop on Rendering). (1996) 21–30.

[8] Jensen, H. and Christensen, P.: Efficient simulation of light transport in scenes with participating media using photon maps. Proceedings of ACM SIGGRAPH. (1998) 311–320.

[9] Kang, B. and Ihm, I. and Bajaj, C.: Extending the photon mapping method for realistic rendering of hot gaseous fluids. Computer Animation and Virtual Worlds. **16(3-4)** (2005) 353–363.

[10] Losasso, F. and Irving, G. and Guendelman, E. and Fedkiw, R.: Melting and burning solids into liquids and gases. IEEE Transactions on Visualization and Computer Graphics. **12(3)** (2006) 343–352.

[11] Min, K. and Metaxas, D.: Realistic Fire Animation based on a Combustion Model. Proceedings of Pacific Graphics 2005. (2005).

[12] Nguyen, D. and Fedkiw, R. and Jensen, H.: Physically based modeling and animation of fire. ACM Transactions on Graphics. **21(3)** (2002) 721–728.

[13] Stam, J.: Stable fluids. Proceedings of SIGGRAPH 1999. (1999) 121–128.

Face Recognition Using 2D and 3D Multimodal Local Features

Ajmal Mian, Mohammed Bennamoun, and Robyn Owens

School of Computer Science and Software Engineering,
The University of Western Australia,
35 Stirling Highway, Crawley, WA 6009, Australia
{ajmal, bennamou, robyn.owens}@csse.uwa.edu.au

Abstract. Machine recognition of faces is very challenging because it is an interclass recognition problem and the variation in faces is very low compared to other biometrics. Global features have been extensively used for face recognition however they are sensitive to variations caused by expressions, illumination, pose, occlusions and makeup. We present a novel 3D local feature for automatic face recognition which is robust to these variations. The 3D features are extracted by uniformly sampling local regions of the face in locally defined coordinate bases which makes them invariant to pose. The high descriptiveness of this feature makes it ideal for the challenging task of interclass recognition. In the 2D domain, we use the SIFT descriptor and fuse the results with the 3D approach at the score level. Experiments were performed using the FRGC v2.0 data and the achieved verification rates at 0.001 FAR were 98.5% and 86.0% for faces with neutral and non-neutral expressions respectively.

1 Introduction

The human face has emerged as one of the most promising biometrics due to its social acceptability and non-intrusiveness. It requires minimal or no cooperation from the subject making it ideal for surveillance and applications where customer satisfaction is important. However, face recognition is very challenging because it is an interclass recognition problem and the distinctiveness of face is quite low compared to other biometrics (e.g. fingerprints) [7]. Moreover, changes caused by expressions, illumination, pose, occlusions and facial makeup (e.g. beard) impose further challenges on accurate face recognition.

A comprehensive survey of face recognition algorithms is given by Zhao et al. [17]. They also categorize face recognition algorithms into holistic, feature-based and hybrid matching algorithms. Holistic matching algorithms basically extract global features from the entire face. Eigenfaces [15] and Fisherfaces [1] are well known examples of holistic face recognition algorithms. Feature-based matching algorithms extract local features or regions such as the eyes and nose and then match these features or their local statistics for recognition. One example of this category is the region-based 3D matching algorithm [13] which matches the 3D pointclouds of the eyes-forehead and the nose regions separately and fuse the results at the score level. Another example is the face recognition using local boosted features [8] which match rectangular regions from facial images at different locations, scales and orientations. Hybrid matching methods use a combination of global and local-features for face recognition e.g. [6].

One limitation of holistic matching is that it requires accurate normalization of the faces according to pose, illumination and scale. Variations in these factors can affect the global features extracted from the faces leading to inaccuracies in the final recognition. Normalization is usually performed by manually identifying landmarks on the faces which makes the whole process semi-automatic. Replacing this manual process by an automatic feature identification algorithm usually deteriorates the final recognition results. Moreover, global features are also sensitive to facial expressions and occlusions. Feature-based matching algorithms have an advantage over holistic matching algorithms because they are robust to variations in pose, illumination, scale, expressions and occlusions.

Multimodal 2D and 3D face recognition provides more accurate results than either of the individual modalities alone [3]. An up to date survey of 3D and multimodal face recognition is given by Bowyer et al. [3] who argue that 3D face recognition has the potential to overcome the limitations of its 2D counterpart however there is a need for better algorithms which are more tolerant to the variations mentioned above. Many 3D face recognition approaches are based on the ICP algorithm [2] or its modified versions. Advantages of ICP based approaches are that perfect normalization of the faces is not required and partial regions of faces can be matched with complete faces. The latter advantage has been exploited to avoid facial expressions [13] and to handle pose variations by matching 2.5D scans to complete face models [10]. The major disadvantage of ICP is that it is an iterative algorithm and is therefore computationally very expensive. Moreover, ICP does not extract any feature from the face and thus rules out any possibility of indexing. Unless another algorithm and or modality is used to perform indexing, ICP based algorithms must perform a brute force matching thereby making the recognition time linear to the gallery size. Selecting expression insensitive regions of the face for matching is a potentially useful approach to overcome the sensitivity of ICP to expressions. However, deciding upon such regions is a problem worth exploring as such regions may not only vary between different persons but between different expressions as well.

In this paper, we present a face recognition algorithm using 2D and 3D multimodal local features. A novel 3D local feature is presented which is a modified version of the tensor representation [11] and extracts features in locally defined 3D coordinates. This makes the feature invariant to pose. Robustness to expressions is achieved by considering only the best predefined m matches. In the 2D domain, the SIFT descriptor [9] is used. SIFTs have mainly been used for pose and scale invariant 2D object recognition which is an intraclass recognition problem. To the best of our knowledge, their use for face recognition has not been thoroughly explored especially using the FRGC v2.0 data (Section 2). In this work, we use the SIFT descriptors for face recognition under illumination and expression variations. The results of the 2D and 3D local features are fused

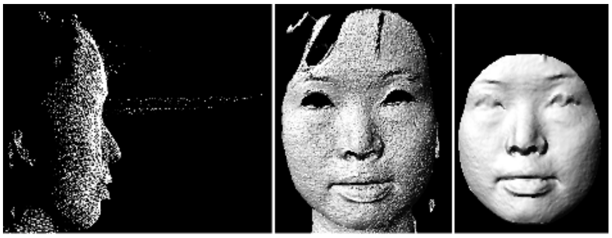


Fig. 1. (Left) A 3D pointcloud of a face shows spikes. (Center) The same face rendered as a shaded view to show noise. (Right) Shaded view after preprocessing.

at the rank level using a confidence weighted sum rule. Preliminary experiments were performed on a *randomly selected* subset of the FRGC v2.0 data [14].

2 Preprocessing the FRGC v2.0 Data

The FRGC v2.0 [14] defines a set of experiments and provides the largest available database for performing each experiment. Of these, only Experiment 3 is relevant to this paper i.e. matching 3D faces (shape and texture) to 3D faces (shape and texture). The FRGC v2.0 data for Experiment 3 consists of multiple 3D faces and their corresponding 2D faces of 466 individuals in the validation set. The database consists of frontal views with minor pose variations and major expression and illumination variations. The individuals are acquired from the shoulder level up (Fig. 2) and therefore a prior step of face detection is needed. Moreover, the 3D data is quite noisy and contains spikes and holes (Fig. 1).

Since preprocessing the data is not at the heart of this paper, we will describe it only briefly. For details, the reader is referred to [12]. A 3D face is automatically detected by locating the nose tip [12]. Next, the region of 3D face inside a sphere of radius r (where $r = 80$ mm) and centered at the nose tip is cropped. The corresponding pixels of the 2D face are also cropped at this stage. The spikes in the 3D face are then removed using a neighbourhood distance constraint and the holes are filled using cubic interpolation. The 3D faces are then median filtered to remove noise. Finally, the pose of the 3D face and its corresponding 2D coloured face is automatically corrected in an iterative algorithm based on the Hotelling transform [12]. The faces are also sampled on a uniform square grid at 1mm resolution during this process. The resultant faces have 161×161 pixels which is reasonable for 2D faces however in the case of 3D faces we delete alternate rows and columns to reduce their size to 80×80 pixels and 2mm resolution.

3 3D Local Features

Our novel local 3D feature is a variant of the tensor representation [11] which quantizes local surface patches of a 3D object into three-dimensional grids defined in locally derived coordinate bases. In [11], we derived the local coordinate

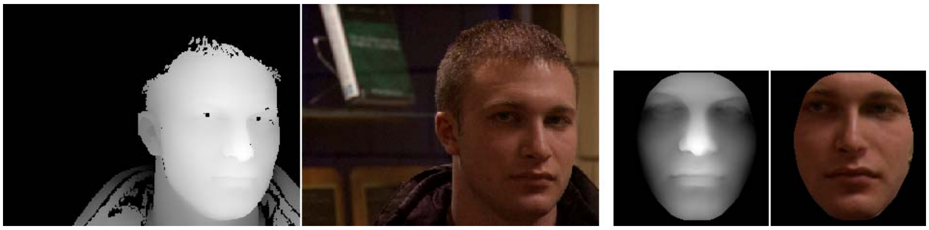


Fig. 2. Illustration of face detection and pose correction of a 3D face and its corresponding coloured texture map

basis from two points and their corresponding normals. In this paper, we define the coordinate basis using a single point in order to avoid the C_2^n (where n is the number of face data points) combinatorial problem [11]. We use a single point and its normal with some additional invariant information to define a local 3D coordinate basis (it is impossible to define a 3D coordinate basis using a single point and its normal alone). Two different types of invariant information were tested for this purpose. The first one was based on the orientation of the SIFT descriptor derived from the 2D face at the corresponding point (details in Section 3.1). The second invariant information used was the location of the nose tip which was detected during the preprocessing (Section 2). Details of this approach are given in Section 3.2.

3.1 Deriving 3D Coordinates from SIFT Orientation and Normal

SIFTs (Scale Invariant Feature Transform) [9] are local descriptors computed at keypoints on a 2D image. These keypoints are extrema in the scale-space and are detected using the difference-of-Gaussian function. Each keypoint is assigned one or more orientations based on the local image gradient. The 2D coloured faces were converted to grayscale images and histogram equalization was used to reduce the effects of illumination before calculating SIFTs [9]. We use the orientations of each SIFT along with the normal of the keypoint calculated from the 3D data in order to define 3D local coordinate bases. The normal of the point, which is calculated by fitting a plane to the neighbouring points within a specified locality, makes the z-axis. The projection of the SIFT orientation on this plane defines the x-axis and the cross product of the z-axis with the x-axis defines the y-axis. Once the 3D basis is defined, the local 3D feature is computed as described in Section 3.3. Multiple orientations at a single point result in multiple 3D bases and hence multiple 3D local features. The downside of this approach is that since SIFTs are stable at only the keypoints, this restricts the number and location of the 3D features. This approach did not give satisfactory results (Fig. 7) indicating that the SIFT keypoint locations were not the best to calculate our 3D features. Another possible reason is that the SIFT orientations were not sufficiently stable to derive unique local 3D coordinate bases.

3.2 Deriving 3D Coordinates from Nose Tip and Normal

In this approach the 3D local coordinate basis at a point is derived using the normal of the point and the location of the nose tip. Since there is a single nose tip, this avoids the C_2^n problem [11] discussed above. Recall that the nose tip is automatically detected during the preprocessing stage (Section 2). The normal is again taken as the z-axis and the cross product of the z-axis with the vector from that point (where the 3D feature is to be calculated) to the nose tip defines the y-axis. The cross product of the y-axis with the z-axis defines the x-axis. Once the 3D basis is defined, the 3D feature is computed as described in Section 3.3. At this stage, we randomly select 300 locations on the face to extract the 3D local features. However, in future we plan to replace this random selection by a more reliable 3D keypoint identification algorithm. This approach gave better results (see Section 4) compared to the first approach probably because the the number and location of the 3D features are not tied up to the SIFTs and some of the 3D local features ended up being selected from regions which are more suitable for 3D features compared to the SIFT keypoint locations. Moreover, the coordinate bases defined using this approach were comparatively more stable.

3.3 3D Local Feature Extraction

Let $\mathbf{P}$ be a $3 \times n$ matrix of the x, y and z coordinates of the pointcloud of a 3D face given by Eqn. 1 (where n is the number of points).

$$\mathbf{P} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \\ y_1 & y_2 & \dots & y_n \\ z_1 & z_2 & \dots & z_n \end{bmatrix} \tag{1}$$

The 3D local feature at a point $\mathbf{p}_i = [x_i \; y_i \; z_i]^\top$ is extracted as follows. First, all points $\mathbf{P}_l$ within a specified neighbourhood l of $\mathbf{p}_i$ are cropped and transformed to the local coordinate basis using Eqn. 2. Where $\mathbf{B}$ is the 3×3 matrix of the locally defined basis.

$$\mathbf{P}'_l = \mathbf{B}(\mathbf{P}_l - \mathbf{p}_i) \tag{2}$$

Eqn. 2 translates $\mathbf{P}_l$ so that $\mathbf{p}_i$ becomes the origin and rotates it so that it is aligned with the local coordinate basis. After rotation, the points in $\mathbf{P}'_l$ may no longer remain uniformly sampled. Therefore, $\mathbf{P}'_l$ is sampled again on a uniform grid in the xy-plane which is essentially the tangent plane used to calculate the normal of $\mathbf{p}_i$. This uniform sampling measures the distance of every sample point in $\mathbf{P}'_l$ to the tangent plane or a local invariant range image at that point. This range image is the 3D local feature at point $\mathbf{p}_i$ and is invariant to pose since it is defined in a local coordinate basis. The value of l decides the degree of locality of the feature and the resolution of the sampling decides the degree of granularity of the feature. Choosing a very high value for l makes the feature sensitive to facial expressions whereas choosing a very low value will make it less descriptive. Similarly, the sampling rate offers a trade off between accuracy and efficiency. We chose $l = 30mm$ and the sampling was done using a 30×30 on the basis of experiments performed on training data.

3.4 Matching 2D and 3D Local Features

It is possible to make the matching process much more efficient by using indexing or hashing. However, we performed a brute force matching in our initial experiments since our aim was to first demonstrate the effectiveness of these features for face recognition. To calculate the similarity between a gallery and probe face, their local features were matched using Euclidean distance. Features with minimum distance were considered as matches. Only one-to-one matches were established i.e. a feature from the gallery face was allowed to be a match to only one probe feature. The similarity score between the two faces was taken as the mean distance between the best predefined m matching pairs of features. This means that some matches were not considered allowing for variations caused in the data e.g. due to illumination and expressions.

Our novel local 3D features are highly descriptive and can find correct matches even in the challenging case of face recognition. Fig. 3 shows the five best matches between a probe with neutral expression and its correct identity in the gallery whereas Fig. 4 shows the five best matches of the same probe with an incorrect identity. The first row corresponds to the 3D local features of the probe whereas the second row corresponds to those of the gallery. Each column represents a matching pair of features with error written between them. Notice that the errors are much lower in the case of the correct identity (Fig. 3). Under non-neutral expressions, the quality of the matches deteriorates however the correct identity still gives much lower error compared to the incorrect identity (Fig. 5 and 6).

3.5 Fusion

The 2D and 3D local feature matching engines each results in a similarity matrix of size $N \times M$ (M is the gallery size and N is the number of probes tested) with negative polarity i.e. a lower value means higher similarity. The matrices are normalized using the min-max rule and then fused using a confidence weighted sum rule. Since each row of a similarity matrix corresponds to an independent recognition trial of a particular probe, the matrices are normalized row wise on the scale of 0 to 1. For each row (or recognition trial), the confidence value is calculated as the ratio of the difference between the best and mean similarity scores to the difference between the second best and mean similarity scores.

4 Results

A single 3D face (neutral expression) per individual, along with its texture, was selected to build a gallery of 466 i.e. the maximum possible using the FRGC v2 data. This was to ensure a thorough and unbiased validation of our algorithm. For each experiment, 200 probes with neutral expression and another 200 with non-neutral expression were randomly selected to ensure that these samples are true representatives of their populations. The negligible difference between the 2D feature performance in Fig. 7 and 8 using different sets of randomly selected

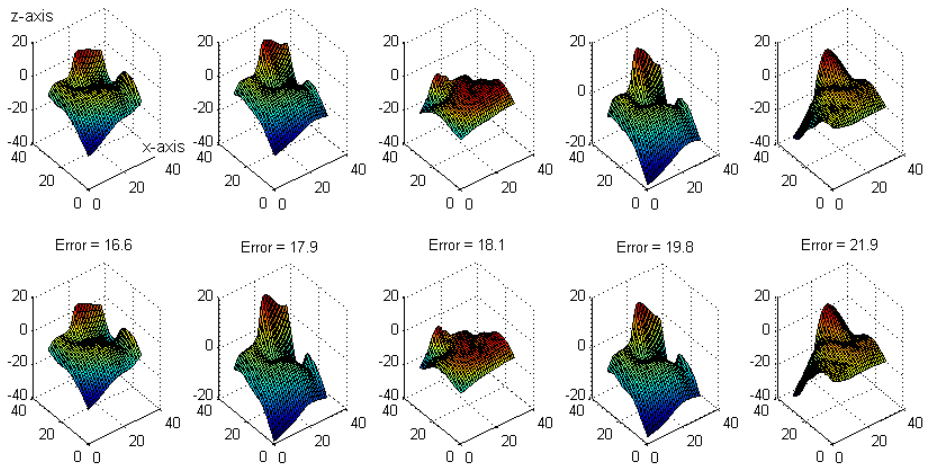


Fig. 3. Best five matches between the 3D local features of a probe (first row) with neutral expression and its correct identity in the gallery (second row). The 3D local features are rendered as 3D surfaces (the nose can easily be noticed as peaks in some of them). Each column shows a matching pair of 3D features with error written between them. Mean error was 39.4 for the best 100 matches in this case.

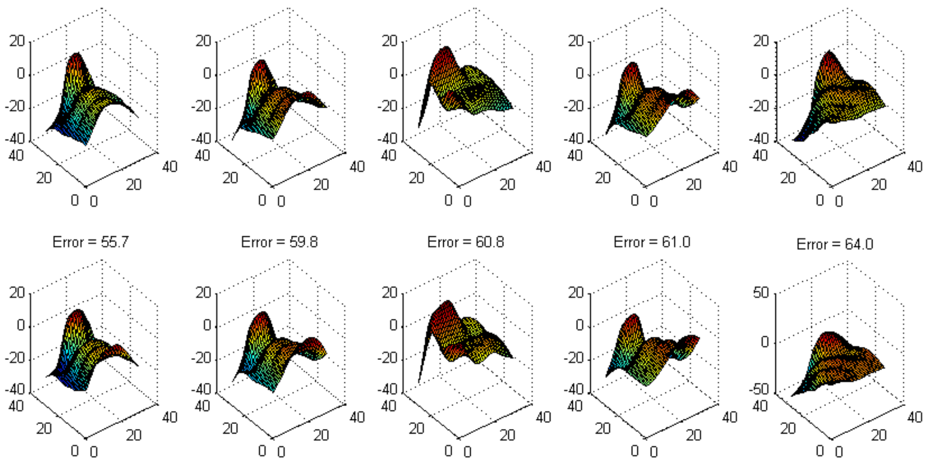


Fig. 4. Best five matches between different identities, under neutral expression, show higher errors compared to Fig. 3. Mean error was 92.8 for the best 100 matches.

faces (neutral expression) supports our claim. All faces were preprocessed (Section 2) and their 2D (SIFT) and 3D local features were calculated and matched.

Fig. 7 shows our results when the local coordinate bases were derived from the SIFT orientations and the point normals. The 3D local features did not perform well in this case due to two possible reasons. One, the location of the SIFT

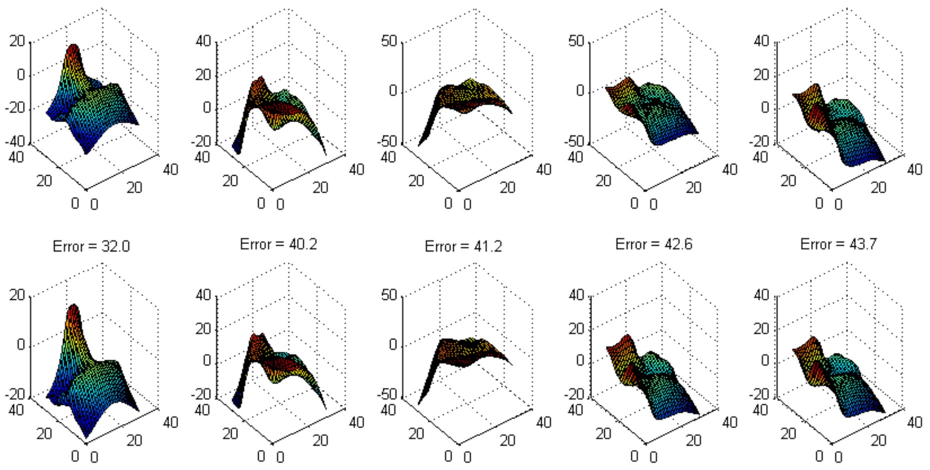


Fig. 5. Best five matches between the 3D local features of a probe (first row) with non-neutral expression and its correct identity in the gallery (second row). Notice that the errors are higher compared to the neutral expression case (Fig. 3) but are still much lower than in the case of an incorrect identity (Fig. 6). Mean error was 93.1 for the best 100 matches in this case.

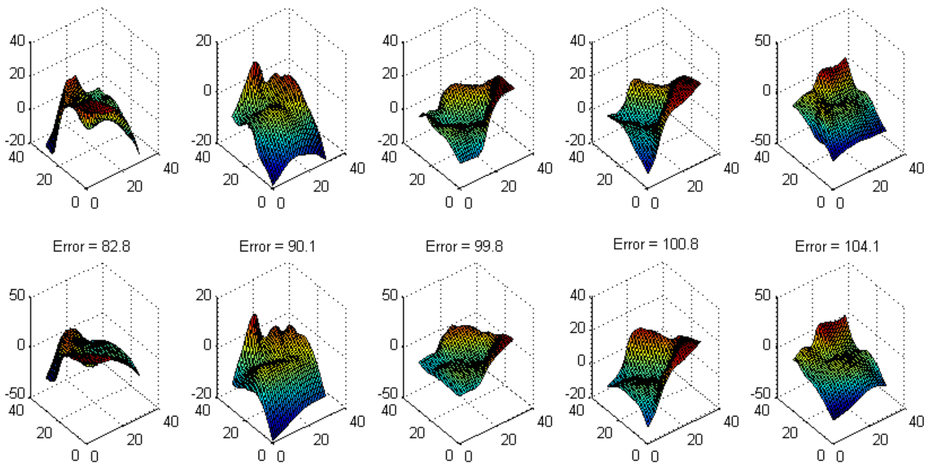
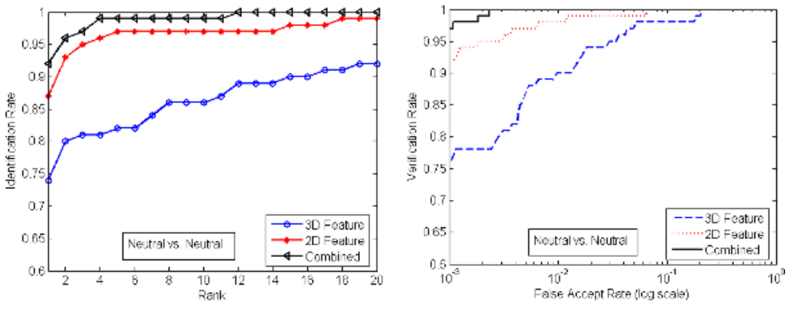


Fig. 6. Best five matches between different identities, under non-neutral expression, show higher errors compared to Fig. 5. Mean error was 156.1 for the best 100 matches.

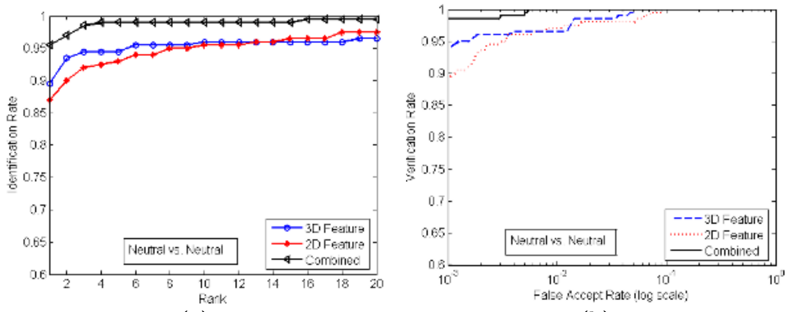
keypoints are not suitable (in terms of descriptiveness) to the 3D local features. Two, the SIFT orientation does not provide stable local coordinates.

In the next experiment, we derived the local coordinate basis from the point normals and the location of the nose tip. Fig. 8 and 9 show our results for probes with neutral and non-neutral expressions respectively. The results were



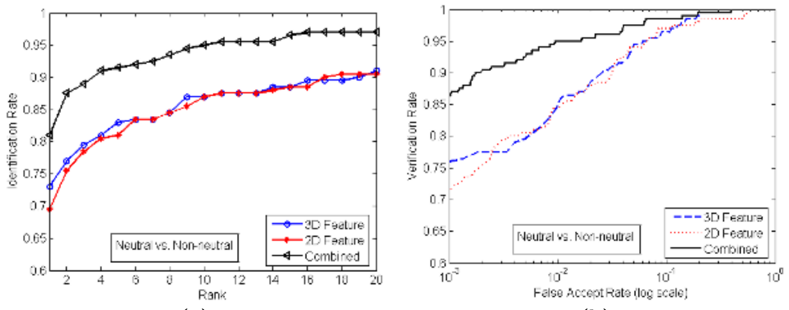
(a) (b)

Fig. 7. (a) Identification and (b) verification performance when the 3D coordinate bases are derived from the point normals and SIFT orientations



(a) (b)

Fig. 8. (a) Identification and (b) verification performance (under neutral expression) when the 3D coordinates are derived from the point normals and the nose tip location



(a) (b)

Fig. 9. (a) Identification and (b) verification performance (under non-neutral expression) when the 3D coordinates are derived from the normals and the nose tip location

very promising in this case as the 3D local features performed much better with individual identification rates of 89.5% and 73.0% for probes with neutral and non-neutral expressions respectively. The verification rates at 0.001 FAR for the same were 94.0% and 76.0% respectively. The 2D local features (SIFT) gave slightly lower but comparable performance to the 3D features. Note that it was not the aim of this paper to provide a true and unbiased comparison of these two features but to demonstrate their use for face recognition in the presence of illumination and expression variations. Fusion of the two features provides a significant improvement in performance with identification rates of 95.5% and 81.0% respectively for probes with neutral and non-neutral expressions. The verification rates at 0.001 FAR for the same were 98.5% and 86.0%.

5 Conclusion

We presented an automatic face recognition algorithm using 2D and 3D local features. We also presented a novel and highly descriptive 3D local feature and demonstrated its performance on a challenging interclass recognition problem i.e. face recognition. We effectively used the SIFT features for face recognition. By combining the 2D and 3D local features, we achieved a significant improvement in performance. Although, our preliminary results show some deterioration under non-neutral expressions, we believe that our 3D feature and recognition algorithm are promising and can be improved or used in conjunction with global features to give more accurate results. Moreover, the combined performance deterioration is significantly lower than that of the individual features. Our analysis show that the failures mainly occur because the 3D features are extracted at inappropriate locations. Therefore, in our future work, we would like to focus on identifying key locations for extracting the 3D local features.

Acknowledgment

We would like to thank the FRGC [14] organizers for providing data and D. Lowe for the SIFT code [9]. This research is sponsored by ARC Grant DP0664228.

References

1. P. Belhumeur, J. Hespanha and D. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection", *IEEE PAMI*, vol. 19, pp. 711–720, 1997.
2. P. J. Besl and N. D. McKay, "Reconstruction of Real-world Objects via Simultaneous Registration and Robust Combination of Multiple Range Images," *IEEE TPAMI*, vol. 14(2), pp. 239–256, 1992.
3. K. W. Bowyer, K. Chang and P. Flynn, "A Survey Of Approaches and Challenges in 3D and Multi-modal 3D + 2D Face Recognition," *CVIU*, vol. 101, pp. 1–15, 2006.
4. C. S. Chua and R. Jarvis, "Point Signatures: A New Representation for 3D Object Recognition," *IJCV*, vol. 25(1), pp. 63–85, 1997.
5. C. Chua, F. Han and Y. Ho, "3D Human Face Recognition Using Point Signatures," *IEEE AMFG*, pp. 233–238, 2000.

6. J. Huang, B. Heisele and V. Blanz, "Component-based Face Recognition with 3D Morphable Models", *AVBPA*, 2003.
7. A. K. Jain, A. Ross and S. Prabhakar, "An Introduction to Biometric Recognition," *IEEE TCSVT*, vol. 14(1), pp. 4–20, 2004.
8. M. Jones and P. Viola, "Face Recognition using Boosted Local Features", *IEEE ICCV*, 2003.
9. D. Lowe, "Distinctive Image Features from Scale-invariant Key Points", *IJCV*, Vol. 60(2), pp. 91–110, 2004 (code available at http://www.cs.ubc.edu.ca/∼lowe/).
10. X. Lu, A. K. Jain and D. Colbry , "Matching 2.5D Scans to 3D Models," *IEEE TPAMI*, Vol. 28(1), pp. 31-43, 2006.
11. A. S. Mian, M. Bennamoun and R. A. Owens, "A Novel Representation and Feature Matching Algorithm for Automatic Pairwise Registration of Range Images", *IJCV*, vol. 66, pp. 19–40, 2006.
12. A. S. Mian, M. Bennamoun and R. A. Owens, "Automatic 3D Face Detection, Normalization and Recognition", 3DPVT, 2006.
13. A. S. Mian, M. Bennamoun and R. A. Owens, "Region-based Matching for Robust 3D Face Recognition", *BMVC*, vol. 1, pp. 199–208, 2005.
14. P. J. Phillips, P. J. Flynn, T. Scruggs, K. Bowyer, J. Chang, K. Hoffman, J. Marques, J. Min and W. Worek, "Overview of the Face Recognition Grand Challenge", *IEEE CVPR*, 2005.
15. M. Turk and A. Pentland, "Eigenfaces for Recognition", *JOCN*, Vol. 3, 1991.
16. C. Xu, Y. Wang, T. Tan and L. Quan, "Automatic 3D Face Recognition Combining Global Geometric Features with Local Shape Variation Information," *IEEE ICPR*, pp. 308–313, 2004.
17. W. Zhao, R. Chellappa, P.J. Phillips, and A. Rosenfeld, "Face Recognition: A Literature Survey", *ACM Computing Survey*, pp. 399-458, 2003.

Adaptive Background Generation for Video Object Segmentation

Taekyung Kim and Joonki Paik

Image Processing and Intelligent Systems Laboratory, Department of Image Engineering, Graduate School of Advanced Imaging Science, Multimedia, and Film, Chung-Ang University, 221 Huksuk-dong, Tongjak-Ku, Seoul 156-756, Korea
kimktk@wm.cau.ac.kr
http://ipis.cau.ac.kr

Abstract. In this paper, we present a novel method for generating background that adopts frame difference and a median filter to sensitive areas where illumination changes occur. The proposed method also uses fewer frames than the existing methods. Background generation is widely used as a preprocessing for video-based tracking, surveillance, and object detection. The proposed background generation method utilizes differences and motion changes between two consecutive frames to cope with the changes of illumination in an image sequence. It also utilizes a median filter to adaptively generate a robust background. The proposed method enables more efficient background reconstruction with fewer frames than existing methods use.

1 Introduction

In order to analyze shape and motion of an object in a video, a background image without objects is a fundamental clue to segment the object. Background generation is widely used in video-based tracking and surveillance systems as a preprocessing method.

An important requirement of robust tracking is the illumination-independent extraction of an object from the video. The information of the extracted object can be reliable when the real motion of the object is found by compensating errors due to the illumination change in the image sequence.

Because of the importance of reliable background generation, current video tracking and surveillance systems adopt various techniques for a reliable object extraction. Object extraction from a video can be performed by using either frame differences or background differences. The frame difference method can be easily adopted in motion analysis. However, it requires an additional post-processing such as tracking. On the other hand, the background difference method that is used in many real-time video systems utilizes differences between the current frame and the background image. Since the background difference-based method is sensitive to shadow and illumination change [1], we propose a robust background extraction method that applies the frame difference method and a median filter to the sensitive areas.

This paper is organized as follows. Section 2 presents background algorithms. In section 3 we describe the proposed background generation method. The experimental results are presented in section 4. Finally, we conclude the paper in section 5.

2 Related Works

A background generation method is one of the preprocessing steps in video tracking systems. The generated background can help to extract an object from video, then the clear object segmentation leads to exact object tracking. The segmentation method for a moving object can be divided into frame difference, background difference, and the average of fixed number of frames [2].

The frame difference method using the differences of frames and generates moving parts. In the frame difference method the background can be removed easily. However, it also removes parts of the moving object [3]. The background difference image and finds moving objects clearly by taking the difference of the input image and the generated background image. On the other hand, it is difficult to get a correct background image [4]. Average filter method [5], which is widely used in the tracking systems for its easy implementation, averages input images pixel by pixel and generates the background from the average values. In the averaging method the quality of the background drops remarkably when the illumination abruptly changes.

Selected object detection and background generation algorithms of significant interests are summarized in Table 1.

Table 1. Classification of background generation algorithm according to environment and approach

Algorithm By	Specific Task	Method Used	Camera View	Sensor	Area
Haritaogu[7]	Surveillance of people	Median Filter	Single	Grayscale	Outdoor
Cucchiara[9]	Object, Ghosts, Shadow detection	Pixel level	Single	Color	Outdoor
Li[10]	Object detection	Pixel level	Single	Color, Grayscale	Indoor, Outdoor
Fang[11]	Object detection	Edge	Stereo	Grayscale	Outdoor
B. Lee[5]	Change detection	Averaging	Single	Grayscale	Indoor, Outdoor
Ren[12]	background subtraction	Mapping of Pixel	Multi	Grayscale	Indoor, Outdoor
Proposed	background generation and object detection	Median Filter and Pixel level	Single	Grayscale	Indoor, Outdoor

2.1 Background Generation and System

An object can be easily extracted by subtracting the input image from a generated background. This subtraction approach can easily detect change in video and is effective in analyzing feature information of an object [4]. More specifically, averaging multiple frames from a video can efficiently remove minor motion and changes, and easily generate a background model [6]. This approach, however, requires too many frames to accumulate, and the extracted background becomes worse as the amount of illumination change increases.

The frame difference method is another simple object extraction method by subtracting two adjacent frames [4]. This can efficiently detect regions that change frame by frame, and be used in many simple background elimination applications.

A sudden change in illumination affects the generated background, and is consequently the major factor that deteriorates the quality of background. Difference between two adjacent frames can be described as

$$Frame(x,y,t) = \left| I(x,y,t) - I(x,y,t-1), \right| \tag{1}$$

Where $I(x,y,t)$ and $I(x,y,t-1)$ represent the current and previous frames respectively. The desirable property of background is to have constant distribution. Based on this context, change in video should not affect the distribution. The W4 [7] algorithm separates objects and background using temporal median filter, and as a result it can provide a constant distribution against illumination changes [8]. This algorithm can also handle fast motion or abrupt change in the image because of the use of median filter. Figure 1 shows a background generation procedure using a median filter.

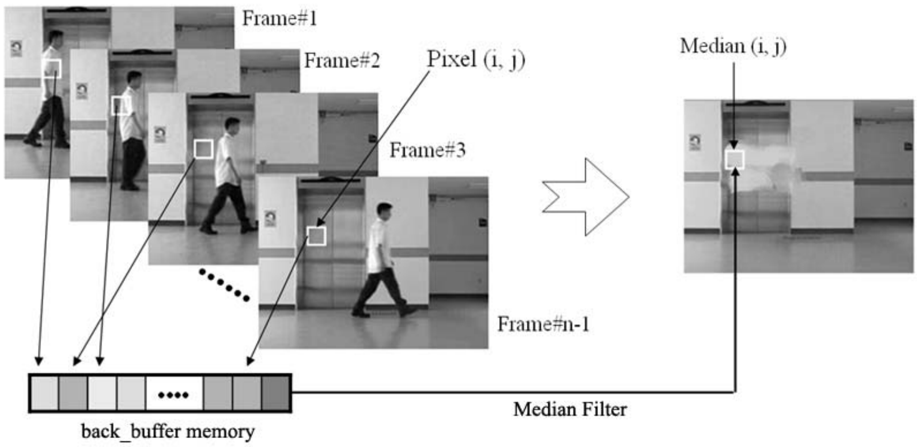
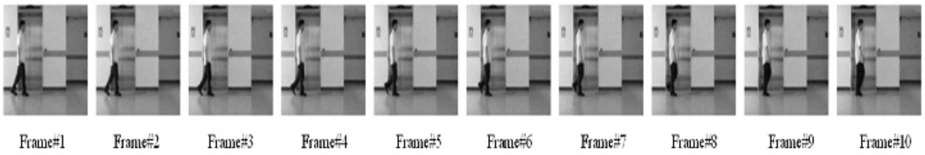
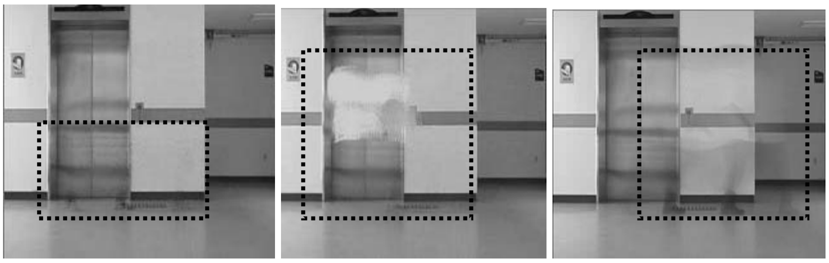


Fig. 1. The background generation using Median Filter

Common drawbacks of existing background generation methods are ghost artifacts that result from frame averaging and noisy the objects. Even the median filter-based



(a) Test of image sequence by in house indoor



(b) Frame difference (c) Median filter (d) Average

Fig. 2. Created background generation model using number of 50 frames

method cannot avoid noisy boundary effect. Various methods are classified and compared in Table 1, and their performance is evaluated in Figure 2.

3 Proposed Method

When the target object and the background image have similar intensity, the object segmentation of existing background generation algorithms becomes sensitive to the threshold value. Thus, the generated background is often incorrect. In the proposed background generation method, the brightness changes of the image sequence are also taken into consideration. In order to prevent sudden changes of the intensity of the generated background, a median filter is used for the final result. Fig 1 shows background generation of the median filter. The procedure of the proposed background generation method can be described as

$$\begin{array}{lll} IF & back\_buffer = |I(x,y,t) - I(x,y,t-1)| \le T, & \\ then & if \quad back\_frame(x,y,t) = Median\_F(back\_buffer) \le T, & (2) \\ else & Median\_F(x,y,t-1), & \end{array}$$

where $I(x,y,t)$ and $I(x,y,t-1)$ respectively represent the current and previous frames. Variable *back_buffer* is Memory Stack (value of frame different), and *T* is the average of the previous frame and *Median_F* is the median filter method. If the intensity of the region that is greater than the threshold value, the region is excluded from the background generation. Fig 3 shows the algorithm flow of the proposed method.

When the brightness difference due to the increase of the number of frames is not considered, a resulting background image has a lot of difference from the input image. If the frame difference is used to segment an object, the value of a pixel can be

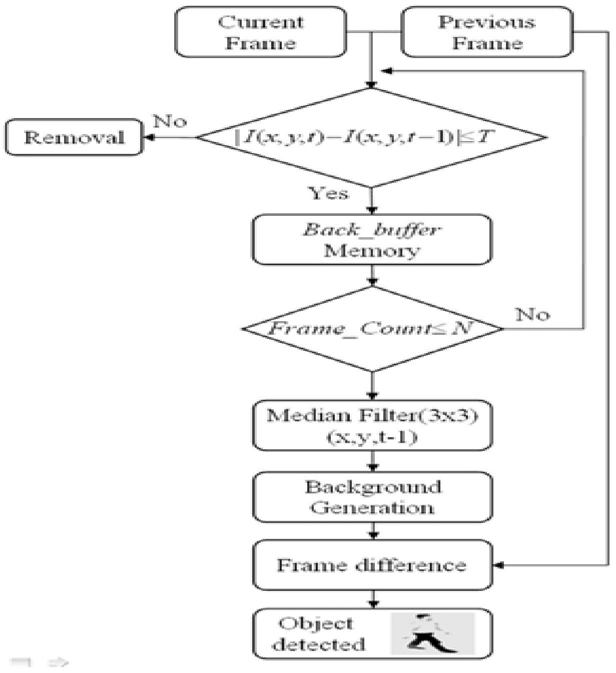


Fig. 3. Structure of the proposed background generation

changed with noise. With a large difference of brightness among the frames, a median filter is used to average the background image. By using a simple subtraction from the current frame, the background generation can be performed as

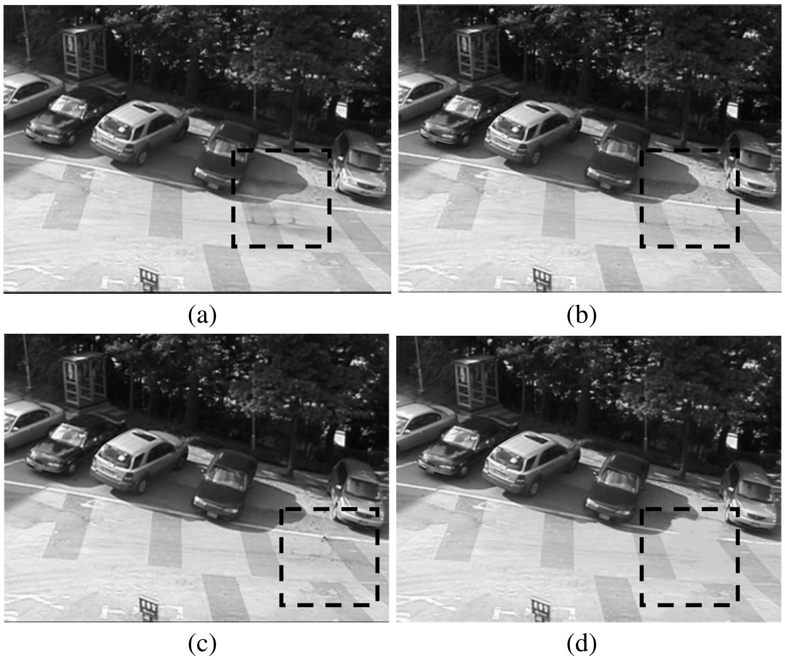
$$O(x,y)=\begin{cases}1, & |I(x,y)-G(x,y)|>T \\ 0, & otherwise\end{cases} \quad , \tag{3}$$

where $I(x,y)$ and $G(x,y)$ represent the current and background generation frames respectively. $O(x,y)$ represents the result of object by the difference Method, and T is the average of the current frame.

4 Experimental Results

In this paper we used standard test image sets provided by PETS-2001 and PETS-2002, in addition to in-house test images. We tested existing methods such as mean, frame difference, and background difference to evaluate and compare the performance of the proposed method. The size of the input image is 320×240. Fig. 4(a) has a big illumination change due to moving clouds, Fig 4. (b) has the reflection problem, and Fig 4.(c) is a nearly ideal image that does not have an illumination change.

Fig. 4. Test Images of (a) In-house and outdoor image, (b) Indoor image provided by PETS-2002, (c) Outdoor image by PETS-2001

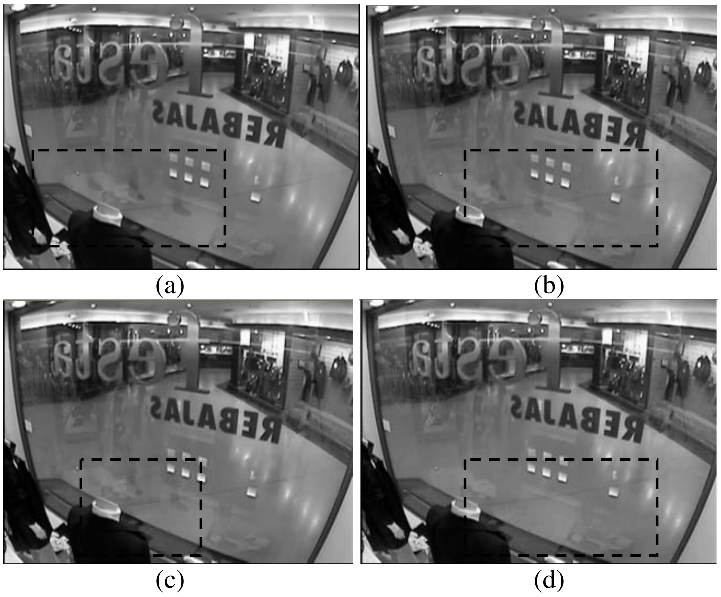


Results of background generation using 40 frames by (a) median filter, (b) edge difference method, (c) frame difference method, and (d) the proposed method.

Fig. 5. Results of four different background generation methods with test image (a)

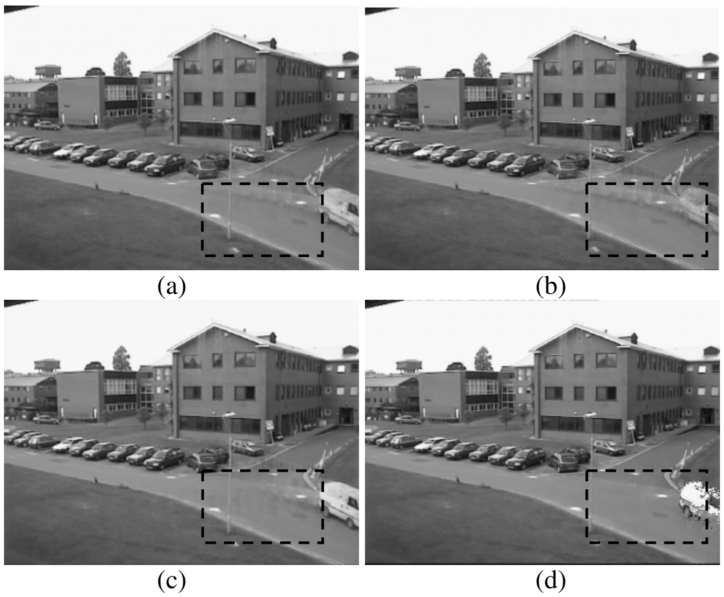
Figures 5, 6, and 7 respectively show the resulting background of four different methods using 40 frames of in-house and outdoor image. Three existing methods generated noisy backgrounds, while the proposed method generated a clear background.

Figure 8 shows the results of object extraction using the proposed background generation. The proposed method can efficiently generate a good quality of noise compared to the existing methods.



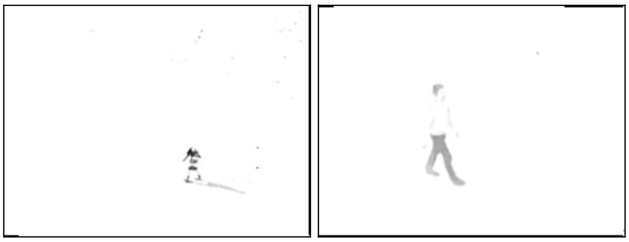
Results of background generation using 40 frames by (a) median filter, (b) edge difference method, (c) frame difference method, and (d) the proposed method.

Fig. 6. Results of four different background generation methods with test image (b)



Results of background generation using 40 frames by (a) median filter, (b) edge difference method, (c) frame difference method, and (d) the proposed method.

Fig. 7. Results of four different background generation methods with test image (c)



Results of object extraction using 35 frames by (a) In-house and outdoor image, (b) Outdoor image by PETS-2002.

Fig. 8. Results of object extraction by the proposed method

Table 2 shows the resulting background of four different methods using 40 frames of all test sequences. We tested three different methods and evaluated peak-to-peak signal-to-noise ratio (PSNR) as

$$PSNR = 10\log_{10}\frac{255^2}{\frac{1}{XY}\sum_{x,y}(I_{x,y} - I'_{x,y})^2}, \tag{4}$$

where *I* represents the original image, *I´* the modified image, and *X* and *Y* respectively the horizontal and vertical sizes of the image.

Table 2. PSNR of the proposed and existing methods

Method	Test image (a) by In-house outdoor	Test image(b) by PETS2002	Test image(c) by PETS2001
Median filter	12.37	20.76	22.39
Average	12.41	19.67	24.15
Edge difference	12.27	17.12	22.80
Frame difference	12.50	19.87	22.92
proposed	**22.57**	**32.34**	**28.83**

5 Conclusions

The change of illumination in the video sequence is one of the important factors that may cause errors in the background generation. Even with many frames in the calculation, the existing methods still some have noise in the generated background due to the accumulated pixel errors. The accumulated errors also affect the separation of the object and background.

In this paper, we propose a background generation algorithm that can resolve ghost effects and illumination change problems by using frame difference and median filter. The experimental results show that the proposed algorithm can efficiently generate a good quality of background with small number of frames compared to the existing algorithm. The computational load of the proposed algorithm, thus, is also small. We present the PSNRs of the existing methods and the proposed method to compare the correctness of the generated background. Possible future research an object extraction technique and moving object tracking based on the proposed background generation model.

Acknowledgment

This research was supported by Korean Ministry of Science and Technology under the National Research Laboratory Project, Korean Ministry of Information and Communication under HNRC-ITRC program at Chung-Ang University supervised by IITA, and by Korean Ministry of Education under the BK21 Project.

References

1. Long, W, Yang, Y, H.: Stationary background generation: An alternative to the difference of two images. Pattern Recognition, Vol. 23. No. 12. pp. 1351-1359. 1990
2. Naohiro, A., Akihiro, F.: Detecting Obstructions and Tracking Moving Objects by Image Processing Technique. Electronics and Communications in Japan, Part. 3. Vol. 2. No. 11. 1999
3. Wixson, L.: Illumination assessment for Vision-based real-time traffic monitoring. Proc. Int. Conf. Pattern Recognition, pp. 56-62. 1996
4. Fathy, M., Siyal, M. Y.: A window-based edge detection technique for measuring road traffic parameters in real-time. Real-Time Imaging, Vol. 1. pp. 297-305. 1995
5. Lee, B., Hedley, M.: Background Estimation for Video surveillance. Int. Image Processing and Computer Vision (ICIAP'01), April 2005
6. Chien, S. Y., Ma, S. Y., Chen, L. G.: Efficient Moving Object Segmentation Algorithm Using Background Registration Technique. IEEE Trans. Circuits and Systems for Video Technology, Vol. 12. No. 7. July 2002
7. Haritaoglu, I.: W4:real-time Surveillance of people and their activates. IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 22. No. 8. 2000
8. Matsushita, Y., Nishino, K., Ikeuchi, K., Sakauchi, M.: Illumination Normalization with Time-Dependent Intrinsic Images for Video Surveillance. IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 26. No. 10. October 2004
9. Cucchiara, R., Costantino, G., Massimo, P., Andrea, P.: Detecting Moving Objects, Ghosts, and Shadows in Video Streams. IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 25. No. 10. October 2003
10. Li, L., Huang, W., Gu, I. H., Tian, Q.: Forground Object Detection in Changing Background Based on Color Co-Occurrence Statistics. Proc. IEEE Workshop on Application of Computer Vision (WACV'02), pp. 269-274. December 2002
11. Fang, Y., Masaki, I., Herthold, B.: Distance/Motion-based Segmentation under Heavy Background Noise. IEEE Intelligent Vehicles Symposium (IV2002), pp. 483-488. July 2002
12. Ren, Y., Chua, C. S., Ho, Y. K.: Statistical background modeling for non-stationary camera. Pattern Recognition Letter, Vol. 24. pp. 183-196. 2003

13. S, J, Kim., S, H, Shin., J, K, Paik.: Real-time iterative framework of regularized image restoration and its application to video enhancement. Real-Time Imaging, Vol. 10. pp. 37-50. 2003
14. A, Koschan., S, Kang, J, K, Paik., B, R, Abidi., M, A, Abidi.: Color active shape models for tracking non-rigid objects, Pattern Recognition Letters, Vol. 24. pp. 1751-1765. 2003
15. Y, Sun., J, K, Paik., A, Koschan., D, L, Page., M, Abidi.: Point fingerprint: A new 3-D object representation scheme, IEEE Trans. Systems, Man and Cybernetics, Part B, Vol. 33. pp. 712-717. 2003
16. Y, Kim., J, Yoo., S, Lee., J, Shin., J, K, Paik., H, Jung.: Adaptive mode decision for H.264 encoder, Electronics Letters, Vol. 40. pp. 1172-1173. 2004
17. S, Kong., J, Heo., B, Abidi., J, K, Paik., M, Abidi.: Recent advances in visual and infrared face recognition – A review, Computer Vision and Image Understanding, Vol. 97. pp. 103-135. 2005

Omnidirectional Camera Calibration and 3D Reconstruction by Contour Matching

Yongho Hwang, Jaeman Lee, and Hyunki Hong

Dept. of Image Eng., Graduate School of Advanced Imaging Science,
Multimedia and Film, Chung-Ang Univ.
hwangyh@wm.cau.ac.kr, leejm@wm.cau.ac.kr, honghk@cau.ac.kr

Abstract. This paper presents a novel approach to both omnidirectional camera calibration and 3D reconstruction of the surrounding scene by contour matching in architectural scenes. By using a quantitative measure to consider the inlier distribution, we can estimate more precise camera model parameters and structure from motion. Since most of line segments of man-made objects are projected to the contours in omnidirectional images, contour matching problem is important in camera recovery process. We propose a novel 3D reconstruction method by contour matching in three omnidirectional views. First, two points on the contour and their viewing vectors are used to determine an interpretation plane equation, and we obtain a contour intersecting both the plane and the estimated patch of the camera model. Then, 3D line segment is calculated from two patches, which is projected to the contour on the third views, and these matching results are used in refinement of camera recovery.

1 Introduction

Camera recovery and 3D reconstruction from un-calibrated images have long been one of the central topics in computer vision. Since the multi-view image analysis is based on establishing correspondence of images, matching features—points, lines, contours—is an important process. When the motion between two images is large, however, the matching problem becomes very difficult.

Omnidirectional camera system is given increasing interest by researchers working in computer vision, because it can capture large part of a surrounding scene. Therefore, wide angle of view often makes it possible to establish many spacious point correspondences which lead to more complete 3D reconstruction from few images. In addition, it is widely used to capture the scene and illumination from all directions from far less number of images.

This paper aims at both the calibration of omnidirectional camera and 3D reconstruction by contour matching in architectural scenes. Contours are more general primitives than points or line segments, and they contain more information about the image. However, most of previous calibration researches of omnidirectional images are based on not the contour correspondence but the feature points such as corners. In addition, although the contour matching using epipolar geometry was proposed, there were few methods to solve this problem in omnidirectional images [1-3].

Straight line features are prominent in most man-made environments, and they provide a great deal of information about the structure of the scene. Additionally, since edge features have more image support than point features, they can be localized more accurately. Since the line segments of man-made objects are projected to contours in omnidirectional images, this paper focused contour matching for 3D reconstruction of the scene structure. The initial estimation of an essential matrix from the point correspondence is used for contour matching. Then, the matched contour is used in refinement of camera recovery. The experimental results showed that the proposed method can estimate omnidirectional camera parameters and achieve more precise contour matching.

The remainder of this paper is structured as follows: Sec. 2 reviews previous studies on calibration of omnidirectional camera and line/contour matching, and the quantitative measure of inlier distributions for omnidirectional camera calibration is discussed in Sec. 3. Sec. 4 presents contour matching using the initial estimation and 3D reconstruction of the line segments in three views. Finally, the conclusion is described in Sec. 5.

2 Previous Studies

Many researches for self-calibration and 3D reconstruction from omnidirectional images have been proposed up to now. Xiong et al register four fisheye lens images to create the spherical panorama, while self-calibrating its distortion and field of view [5]. However, camera setting is required, and the calibration results may be incorrect according to lens because it is based on equi-distance camera model. Sato et al simplify user's direct specification of a geometric model of the scene by using an omnidirectional stereo algorithm, and measure the radiance distribution. However, because of using the omnidirectional stereo, it is required in advance a strong camera calibration for capturing positions and internal parameters, which is complex and difficult process [6].

Although previous studies on calibration of omnidirectional images have been widely presented, there were few methods about estimation of one parametric model and extrinsic parameters of the camera [7~9]. Pajdla et al just metntioned that one parametric non-linear projection model has smaller possibility to fit outliers, and explanied that simultaneous estimation of a camera model and epipolar geometry may be much affected by sampling corresponding points between a pair of the omindirectional images [10]. However, it requires further consideration of selecting more proper inlier set to overcome false point matching problem. In addition, using contour, which are more general primitives than points or line segments, makes it possible to achieve 3D reconstruction robust to large motion and occlusions. This paper presents a calibration algorithm for the omnidirectional camera by considering the inlier distribution, and 3D reconstruction by contour matching in three views.

Previous studies on contour/line matching are classified into two classes according to whether the geometric constraints were used. Some researches using the multi-view geometry enables to cope with occlusion of the contour points by large camera motion on several views, in spite of high computational complexity [1, 2]. Although the contour matching using epipolar geometry was proposed, there were few methods to solve the contour matching problem in omnidirectional images based on the estimated camera information. A line-photogrammetric mathmatical model for 3D reconstruction was built with image line observations and object parameters in the form of the coordinates of object points and the parameters of object planes [4]. Because in this model the orientation of the images is assumed to be approximately known and coplanarity properties of the observed image lines are used, it has many constraints in real applications. On the contrary, another approach assumes generally the continuity of contours in bright and shape over successive frames [3], but the smoothness constraints may be often violated due to various lighting effects.

3 Omnidirectional Camera Model Estimation

3.1 One-Parametric Projection Model

The camera projection model describes how 3D scene is transformed into 2D image. The light rays are emanated from the camera center that is the camera position, and determined by a rotationally symmetric mapping function f .

$$f(u,v) = f(\mathbf{u}) = r / \tan\theta \tag{1}$$

where, $r = \sqrt{u^2 + v^2}$ is the radius of a point (u, v) with respect to the camera center and θ is the angle between a ray and the optical axis.

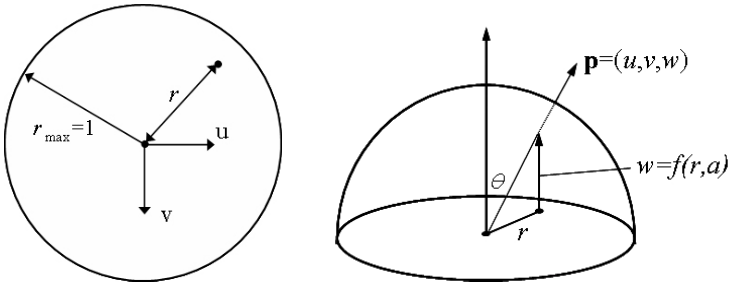


Fig. 1. Image formation and calibration of one-parametric omnidirectional camera model

The mapping function f has various forms by lens construction [11]. We derive an one-parametric non-linear model for Nikon FC-E8 fisheye converter as follows:

$$\theta = \frac{ar}{1 + r^2\left(ar_{\max} / \theta_{\max} - 1\right)}, \tag{2}$$

where a is a parameter of the model. On the assumption that the maximal view angle θ_{max} (=1.597 rad) is known, the maximal radius r_{max} (normalized as 1) corresponding to θ_{max} can be obtained from the view field image.

In order to estimate one parametric non-linear projection model, we use two omnidirectional images with camera direction and translation. Corresponding points between two views are established by MatchMover pro3.0 [12] and then, the essential matrix is estimated by quadratic eigenvalue problem and epipolar geometry [8].

3.2 Camera Pose Estimation Using Inlier Distribution

One of the main problems is that the essential matrix is sensitive to the point location errors. In order to cope with the unavoidable outliers inherent in the correspondence matches, our method is based on 9-points RANSAC that calculates the point distribution for each essential matrix [13]. Since the essential matrix contains relative orientation and position of the camera, the inliers represent the depths of the scene points and change of the image by camera motion. By considering the point distribution, we can select effectively the inlier set that reflects the scene structure and the camera motion, so achieve more precise estimation of the essential matrix.

The standard deviation of the point density in the sub-region and that in an entire image can be used to evaluate whether the points are evenly distributed. First, 3D patches are segmented by the same solid angle in the hemi-spherical model and then they are projected onto the image plane as shown in Fig. 2.

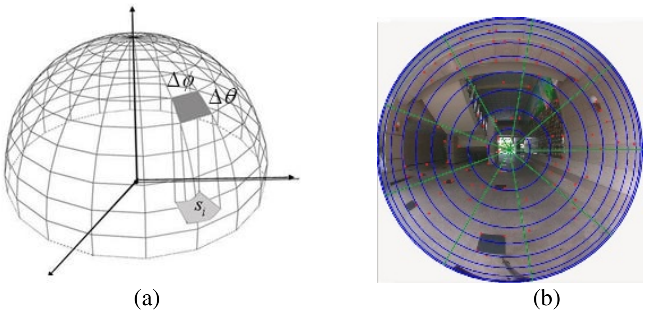


Fig. 2. Segmented sub-regions (a) Segmented 3D patch by uniform solid angle in hemi-spherical model (b) 2D projected sub-regions and inlier set

$$\Delta\theta = 0.5\pi / int(\sqrt{N}),\ \ \Delta\phi = 2\pi / int(\sqrt{N}) \tag{3}$$

where N is the number of the inliers, and $int(\cdot)$ means conversion to integer. The proposed method computes the standard deviation of two densities that represents a degree of the point distribution in each sub-region relative to the entire. The obtained information is used as a quantitative measure to select the evenly distributed point sets. The standard deviation of the point density is defined as:

$$\sigma_p = \sqrt{\frac{1}{N_S}\sum_{i=1}^{N_S}\left(P_{S_i} - \frac{N}{N_S}\right)^2} \tag{4}$$

where N_S is the number of sub-regions, N and P_{Si} are the number of inliers and that in the i-th sub-region, respectively.

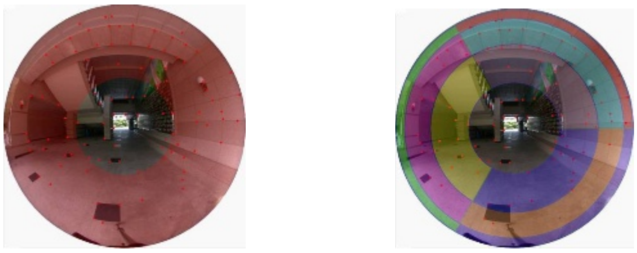


Fig. 3. Segmented regions for selecting inlier sets (a) Previous method (b) Proposed method ($N_S = 9$)

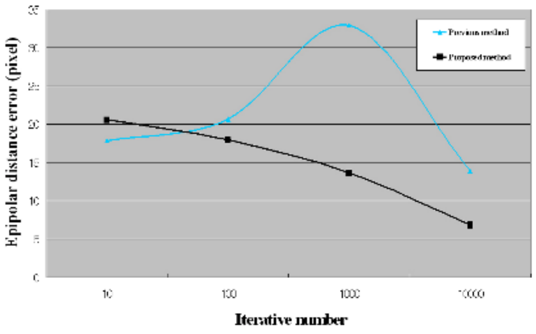


Fig. 4. Experimental results on the omnidirectional image pair

The proposed method chooses each inlier set by using the standard deviation of distribution of each inlier set by Eq. (4), and we find the inlier set with the least standard deviation. In the final step, we estimate the essential matrix from the selected inlier set by minimizing the cost function that is the sum of the distance error of the inliers.

Our method is compared with the previous method using 9-poins RANSAC. The regions for selecting inlier sets are showed in Fig. 3. Since points near the camera center have no special contribution to the final fitting, the center region is omitted [10]. Fig. 4 shows the computed epipole distance error. The results show our method gives relatively better results over the previous method as the iteration number increases.

4 3D Reconstruction by Contour Matching

In the section 3 we estimated the omnidirectional camera with one parametric model and the essential matrix simultaneously. That means we can obtain just the unit vector for the camera translation. Therefore, because the real amount of translation is not known, it is difficult to make a precise registration of camera recovery from multi-view omnidirectional images. This section presents 3D reconstruction method by

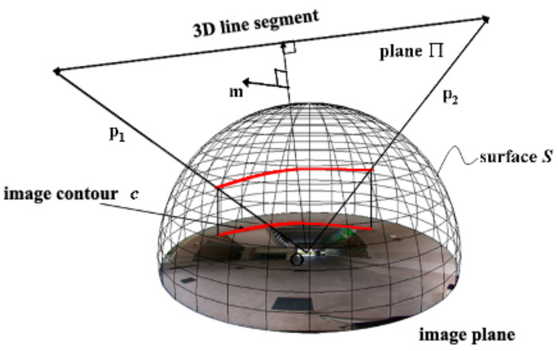


Fig. 5. 3D line segment projected as the contour on the image plane

contour matching in three omnidirectional views (the base, reference and camera view). In this process, the estimated camera motions and 3D scene structure can be integrated.

Fig. 5 shows how 3D straight line segment with a begin point and an end is projected as the contour on the image plane. The line segment and the camera center **O** (0, 0, 0) define an interpretation plane Π whose the normal vector **m** = (m_x, m_y, m_z) obtained from the direction vectors (**p₁**, **p₂**) of the begin and the end through their cross production ($\mathbf{p}_1 \times \mathbf{p}_2$). The image contour ***c*** is the projection of the intersection between the plane Π and the surface ***S***, which represents the omnidirectional camera model. The plane Π is defined using an arbitrary point **p** (x, y, z) and the center of the camera $\mathbf{p}_0$ as follows:

$$\mathbf{m} \cdot (\mathbf{p} - \mathbf{p}_0) = 0, \tag{5}$$

where $\mathbf{p}_0$ is the camera center because we choose the usual canonical camera for the first view.

By executing the dot product in Eq. (5), the plane equation of Π is simplified with respect to z as follows:

$$z = ux + vy\ , \quad u = -m_x / m_z\ , v = -m_y / m_z\ . \tag{6}$$

Before introducing the curve intersecting both the plane and the estimated surface of the camera model, we first obtain the curve intersecting both the plane and the unit sphere for computational simplicity, and rearrange it with respect to y as follows:

$$\begin{aligned} ux + vy &= \sqrt{1 - x^2 - y^2}\,, \\ (v^2+1)y^2 + 2uvxy &+ (u^2+1)x^2 - 1 = 0 \end{aligned} \tag{7}$$

To evaluate the curve intersecting both the plane and the estimated surface of the camera, we need to determine θ and r, because the derived camera model is defined with respect to these variables as Eq. (2). We can compute θ from the unit sphere, and r is obtained using Eq. (2) as follows:

$$\theta = \tan^{-1}\left(\frac{\sqrt{x^2+y^2}}{z}\right), \quad r = \frac{a - \sqrt{a^2 - 4\left(\frac{ar_{\max}}{\theta_{\max}-1}\right)\theta^2}}{2\left(\frac{ar_{\max}}{\theta_{\max}-1}\right)\theta} \tag{8}$$

When r is computed, we can draw the contour on the omnidirectional image by using Eq. (9). The curve intersecting both the plane and the camera model surface is projected as the contour onto the image plane. 2D image contour is effectively used to determine the interested region for contour matching of 3D line segment.

$$y = \pm\sqrt{r^2 - x^2} \tag{9}$$

In order to register camera motions, we multiply the unit translation vector obtained from the essential matrix by a scaling factor, and determine the camera position of the reference view. As shown in Fig. 6, the reference scale s_{ref} is fixed as a constant, and then the equation of 3D infinite line can be determined by finding the intersection between the plane Π_{b} and Π_{ref} as follows:

$$\frac{x-[m_{\mathrm{b,y}}d/(m_{\mathrm{b,y}}m_{\mathrm{ref,x}}-m_{\mathrm{b,x}}m_{\mathrm{ref,y}})]}{m_{\mathrm{x}}} = \frac{y-[m_{\mathrm{b,x}}d/(m_{\mathrm{b,y}}m_{\mathrm{ref,y}}-m_{\mathrm{b,y}}m_{\mathrm{ref,x}})]}{m_{\mathrm{y}}} = \frac{z}{m_{\mathrm{z}}}, \tag{10}$$
$$d = -s_{\mathrm{ref}}(m_{\mathrm{ref,x}}\hat{t}_{ref,\mathrm{x}} + m_{\mathrm{ref,y}}\hat{t}_{ref,\mathrm{y}} + m_{\mathrm{ref,z}}\hat{t}_{ref,\mathrm{z}})$$

where $\mathbf{m_b}$, $\mathbf{m_{ref}}$ and $\mathbf{m}$ are normal vectors of plane Π_{b} and Π_{ref}, and their cross product, respectively.

The begin point and the end of 3D line segment are used to calculate its line equation. Because the lines (L_1 and L'_1, L_2 and L'_2) are intersected in 3D space, we can compute two points of 3D line by using the center of each camera model and the direction vectors ($\bar{d}_1$ and $\bar{d}'_1$, $\bar{d}_2$ and $\bar{d}'_2$) meeting at two points (P_1 and P_2). The equation of line of endpoints P_1 from the center of the surface is evaluated as following:

$$\begin{aligned} &L_1: \ x = \bar{d}_{1x}t_1, \quad y = \bar{d}_{1y}t_1, \quad z = \bar{d}_{1z}t_1 \\ &L'_1: \ x = t_{ref,x} + \bar{d}'_{1x}s_1, \quad y = t_{ref,y} + \bar{d}'_{1y}s_1, \quad z = t_{ref,z} + \bar{d}'_{1z}s_1, \quad t_1, s_1 \in R \end{aligned} \tag{11}$$

By using first two terms of Eq. (11), we compute t_1 and s_1 as following

$$t_1 = \frac{t_{ref,x}\bar{d}'_{1y} - t_{ref,y}\bar{d}'_{1x}}{\bar{d}_{1x}\bar{d}'_{1y} - \bar{d}'_{1x}\bar{d}_{1y}}, \quad s_1 = \frac{t_{ref,x}\bar{d}_{1y} - t_{ref,y}\bar{d}_{1x}}{\bar{d}_{1x}\bar{d}'_{1y} - \bar{d}'_{1x}\bar{d}_{1y}} \tag{12}$$

When substituting the computed t_1 to L_1 or s_1 to L'_1, we can obtain the point (P_1) on 3D line segment. Similarly, P_2 can be computed by using the L_2 and L'_2. By using the

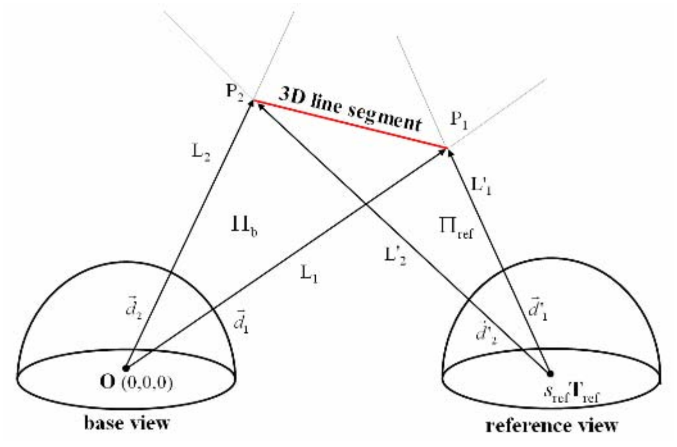


Fig. 6. Projection of 3D line segment onto two omnidirectional images

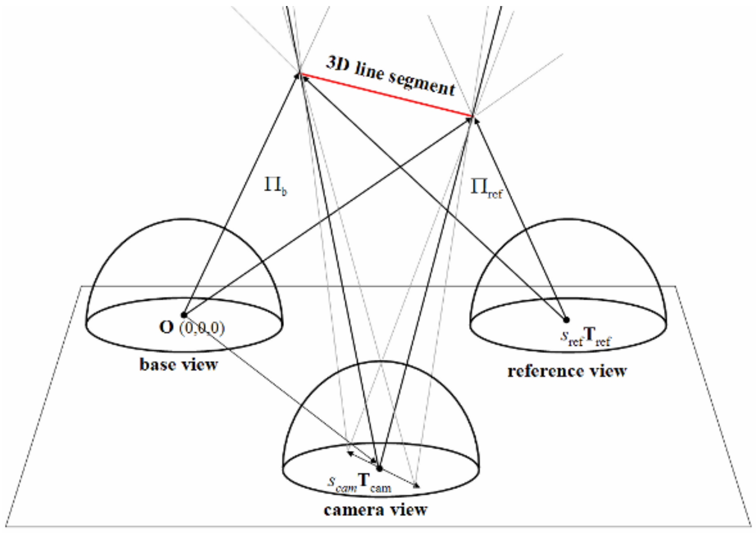


Fig. 7. Projection of 3D line segment onto multi-view omnidirectional images

computed information of 3D line segment, we calculate the relative position of the curve projected onto the third image. From three points P_1, P_2 and $S_{cam}\mathbf{T}_{cam}$ we evaluate the interpretation plane equation. In addition, we compute the curve intersecting both the plane with the surface, which is applicable to the guided contour matching based on the geometrical information as well as refinement of the multi-view estimation.

Fig. 8 shows a typical situation in the image plane, and we define the contour distance error *D* as follows:

$$D = \sum_{i=1}^{n} D_i, \quad D_i = \int_0^1 \left\| c_i(s) - c_i(t) \right\| ds \tag{13}$$

where i and n are the subscribe of the contour and the number of contours, and $t(s)$ are monotonic non-decreasing function with $t(0)=0$ and $t(1)=1$.

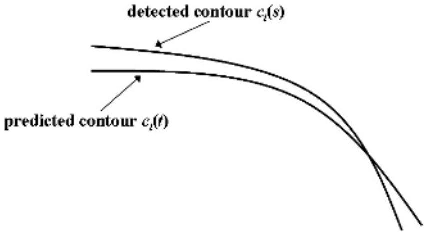


Fig. 8. Predicted and detected contours in the image

16896 and 18768 contours in two views were extracted by Canny's [15], respectively. Among them, the contours are discarded when the distance errors are higher than the constant value. The contour selection process is continued until the average distance is less then a predefined threshold value. Over 76 contour pairs over three views were remained finally, and then the initial camera pose was refined by the translation vector's scale which has minimum D.

5 Conclusions

This paper presents a novel approach to both omnidirectional camera calibration and 3D reconstruction by contour matching in architectural scenes. Using a quantitative measure to consider the inlier distribution makes it possible to estimate more precise camera parameters. In addition, we present a novel mathematical method for contour matching over three views by using the estimated camera information. The matched contour is applicable to the guided contour matching based on the geometrical information as well as refinement of the multi-view estimation. Further study will include reconstruction of 3D scene structure and evaluation of 3D positions of the light source to generate photorealistic images.

Acknowledgments. This research was supported by Seoul Future Contents Convergence (SFCC) Cluster established by Seoul Industry-Academy-Research Cooperation Project.

References

1. J. Han and J. Park, "Contour matching using epipolar geometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.22, no.4, pp.358-370, 2000.
2. C. Schmid and A. Zisserman, "Automatic line matching across views," *In proc. of the IEEE Conference on Computer Vision and Pattern Recognition* (1997), pp. 666-672.

3. R. N. Strickland and Z. Mao, "Contour motion estimation using relaxation matching with a smoothness constraint on the velocity field," *Computer Vision, Graphics, and Image Processing: Image Understanding*, vol.60, no.2, pp.157-167, 1994.
4. F.A. van den Heuvel, "A line-photogrammetric mathematical model for the reconstruction of polyhedral objects," *Proceedings of SPIE*, vol.3641, pp.60-71, 1999.
5. Y. Xiong and K. Turkowski, "Creating image based VR using a self-calibrating fisheye lens," *Proc. of Computer Vision and Pattern Recognition*, pp.237-243, 1997.
6. I. Sato, Y. Sato, and K. Ikeuchi, "Acquiring a radiance distribution to superimpose virtual objects onto a real scene," *IEEE Trans. on Visualization and Computer Graphics*, vol.5, no.1, pp.1-12. 1999.
7. R. Bunschoen and B. Krose, "Robust scene reconstruction from an omnidirectional vision system," *IEEE Trans. on Robotics and Automation*, vol.19, no.2, pp.358-362, 2003.
8. B. Micusik and T. Pajdla, "Estimation of omnidiretional camera model from epipolar geometry," *Proc. of Computer Vision and Pattern Recognition*, pp.485-490, 2003.
9. B. Micusik, D. Martinec, and T. Pajdla, "3D Metric reconstruction from uncalibrated omnidirectional Images," *Proc. of Asian Conf. on Computer Vision*, pp.545-550, 2004.
10. B. Micusik and T. Pajdla, "Omnidirectional camera model and epipolar estimateion by RANSAC with bucketing," *IEEE Scandinavian Conf. Image Analysis*, pp. 83-90, 2003.
11. J. Kumler and M. Bauer, "Fisheye lens designs and their relative performance," http://www.coastalopt.com/fisheyep.pdf.
12. http://www.realviz.com
13. R. Hartley and A. Zisserman: Multiple View Geometry in Computer Vision, Cambridge Univ., 2000.
14. C. J. Taylor, D.J. Kriegman, "Structure and motion from line segments in multiple images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.17, no.11, pp.1021-1032, 1995.
15. J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.8, no.6, pp.679-698, 1986.

Real-Time GPU-Based Simulation of Dynamic Terrain

Anthony S. Aquilio, Jeremy C. Brooks, Ying Zhu*, and G. Scott Owen

Department of Computer Science
Georgia State University
Atlanta, Georgia, USA
yzhu@cs.gsu.edu

Abstract. Although a great deal of research effort has been devoted to the realistic and efficient rendering of terrain data, the existing research mainly focuses on displaying static terrain. The realistic simulation of dynamic terrain, including vehicle-terrain interactions, is absent in most 3D graphical applications. In this paper, we present a new GPU-based algorithm for dynamic terrain simulation. A key component of this algorithm is a Dynamically-Displaced Height Map (DDHM) which is generated and manipulated on the GPU. In addition, our method achieves real-time performance by using new features of the latest graphics hardware and shader technology. We demonstrate our algorithm by simulating a ground vehicle traveling on soft terrain. Our algorithm is particularly useful for ground based visual simulation applications as well as 3D games.

1 Introduction

Terrain visualization has been an essential element in applications such as scientific visualization, ground based training, civil engineering simulation, and 3D games. Many of these applications involve simulating ground vehicles traveling on soft terrain. Although a great deal of research effort has been devoted to the realistic and efficient rendering of terrain data, the existing research mainly focuses on static terrain. As a result, the realistic simulation of vehicle-terrain interaction is absent in most 3D graphical applications. One example of such vehicle-terrain interaction is the tracks left by an off-road vehicle on soft terrain, such as snow or sand. Such visual cues are not only important for improving the realism of the simulation, but also are useful for training purposes.

In this paper, we present a real-time Graphics Processing Unit (GPU)-based algorithm for dynamic terrain simulation. We define dynamic terrain simulation as the simulation of terrain deformation during its interaction with other entities. The main focus of this paper is on simulating terrain-vehicle interaction. Here a vehicle is regarded as any mobile entity regardless of the mechanical, electrical, or biological framework facilitating the movement. Thus our definition of vehicle includes both mechanical vehicles and living creatures.

A major design goal of our algorithm is to offload terrain manipulation to the GPU as much as possible. As a result, our algorithm runs almost entirely on the GPU. The

* Corresponding author.

key components of this algorithm are a Dynamically-Displaced Height Map (DDHM) and its corresponding offset map, which are both generated and manipulated on the GPU. In addition, our algorithm uses new features of the latest graphics hardware and shader technology, such as vertex texture based displacement mapping and framebuffer objects, to achieve real-time performance.

Our algorithm is particularly useful for ground based visual simulation applications as well as 3D games. We demonstrate our algorithm by implementing a real-time driving simulation in which a physics based automobile leaves tracks on soft terrain.

The rest of the paper is organized as follows. In section 2 we discuss related work in dynamic terrain simulation. In section 3, we discuss our algorithm in detail. Implementation and experiments are discussed in section 4. We analyze our method in section 5. Our conclusion and future work are described in section 6.

2 Related Work

Dynamic terrain systems produce surface deformations that are used to visually convince the observer of terrestrial interaction. In general, there are two approaches to simulate dynamic terrain: physics-based and appearance-based approaches.

Physics-based solutions simulate terrain surfaces using the physical characteristics of their natural embodiment. The simulation model is derived using Soil Mechanics and Geotechnical Engineering to handle the management and execution of surface modifications. The tradeoff for the high quality realism is the increased computational cost and operational complexity due to the underlying physics simulation.

Li and Moshell presented a physics-based solution for dynamic terrain [6]. Their physics-based method accurately simulates the local erosion process, which allows for the visual presentation of piling, cutting, and moving of granular soil. The simulation model decomposes the soil into particle masses that, collectively, have a shear strength and shear stress for use in the erosion process. The model lacks a representational soil composition, thereby causing it to be insufficient for simulating soil compression.

Chanclou et al. propose a different method for simulating soil dynamics in a visual system [7]. The method is a two phase approach that treats the terrain surface as an elastic sheet. The first phase is a large-scale deformation that imitates soil compaction and displacement. The second phase performs small-scale refinement that performs local erosion to eliminate unrealistic pilings. Unfortunately, the method is very slow and not intended for use in interactive, real-time systems.

Appearance-based methods attempt to create convincing visuals without the imposition of using a physically accurate model. In an effort to improve performance and ease of use, appearance-based techniques will fabricate parameterizations and invent functional constructs to achieve visually-convincing terrain surface modifications. The simplifications tend to constrain the simulation to a limited range of soil types and composition. For instance, many video game systems use decals to achieve a sub-par simulation of tire-soil interaction.

Sumner et al. present an appearance-based method for animating sand, mud, and snow [8]. It is a four-step process controlled with five parameters. Although the method can achieve a reasonable visual display, it can only produce a smooth deformation; rendering it unable to produce well-formed surface changes. Also, the authors note that rendering parameters had to be hand-tailored in order to achieve optimal visuals.

Onoue and Nishita [9] improved upon the work of Sumner et al. by incorporating a Height Span Map, which allows granular materials to be piled onto the top of objects. In addition, the direction of impacting forces is taken into consideration during the displacement step, thereby improving the visual realism in animations. However, the model still produces smoothed deformations and does not maintain a record of soil composition.

A major limitation of the existing dynamic terrain algorithms is their performance. Most of them are not suitable for real-time interactive applications. None of them takes advantage of the powerful GPU. The work presented here, an appearance based method, tries to address this problem by using features of the latest graphics hardware and shader technology. Our method achieves expeditious throughput and performance suitable for interactive real-time systems.

Unlike many of the existing methods our solution does not operate under the assumption of a granular material model, therefore our method is suitable for simulating clay-like ground materials. In general, our method eliminates the limitation that prescribes smoothed deformations, to provide a new class of simulation possibilities, where vehicles, munitions, or biological entities can influence their surroundings in a natural and realistic manner.

3 Terrain Deformation Algorithm

3.1 Dynamically-Displaced Height Map (DDHM)

A common data structure for storing terrain elevation data is a height map, which has been shown to outperform other terrain mesh types [10]. A height map is composed of a rectilinear grid of evenly-distributed elevation points. The data layout allows for storing the elevation samples in 2D image space, an exploitable property. The elevation value can be decoded from the height map into a 3D positional vertex value (x, y, z) as follows:

$$x = k_x \times x$$
$$y = k_y \times S_{xz}$$
$$z = k_z \times z$$

where S is the input set of elevation data and k is a positive, non-zero scalar value.

Dynamic terrains are unique in their ability to be modified at runtime. In our algorithm, the dynamic terrain surface is stored as a *Dynamically-Displaced Height Map* (DDHM). The DDHM is generated from the input set of elevation data and, subsequently updated each frame to reflect topographical changes. In this manner, the terrain is incrementally modified to produce a history of activity, resulting in a record that can be displayed or queried in successive frames.

3.2 Algorithm Overview

The following steps are performed to accomplish the goal of deforming the terrain.

1. Initialization: Transform the input height map to a DDHM.
2. Calculate the terrain elevation offset values in a first render pass.
3. Deform the terrain in a second render pass, using the DDHM from step 1 and the offset map generated in step 2.
4. Repeat step 2 and 3 for each frame until the end of simulation.

Note that steps 1 through 4 are performed entirely on the GPU. The following sections cover each step in greater detail.

3.3 Initialization

The terrain elevation data is stored in a height map, a rectilinear grid of data samples. This allows the input data sample to exist in 2D image space S . A Dynamically-Displaced Height Map (DDHM) is also a rectilinear grid of data values that exists in 2D image space V . A DDHM is generated through a special rendering pass. First, the terrain elevation data is loaded. Then the view parameters are configured such that the view volume encapsulates the area of the terrain that is to be deformed. The area to be deformed can be extracted by examining the bounding volume of the vehicles (or other objects) intersecting the terrain surface. The camera is then positioned and oriented such that the viewing direction is perpendicular to the ground plane and directly beneath the center of the subarea to be deformed (Figure 1). After that, the terrain is rendered into the render target using the bottom-up view of the camera. The DDHM is defined as the contents of the target buffer and is held in the graphics memory for the rest of program execution.

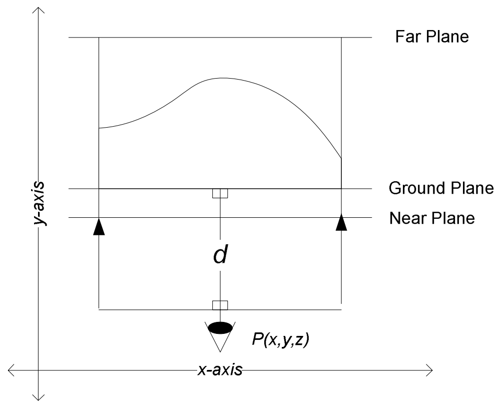


Fig. 1. The camera configuration for generating the DDHM

Let U_S represent the height map's elevation data and let U_V represent the corresponding data in the DDHM. The above rendering pass can be treated as a transformation function, $U_S \rightarrow U_V$, such that every data sample maps in a one-to-one relationship. This transformation function can achieve up sampling and down sampling by using a one-to-many or many-to-one function, respectively. Resampling elevation data offers the opportunity to alter the granularity of the deformation, which can be used to throttle the simulation and visuals as deemed necessary by the application.

In practice, it may be necessary to use the elevation data of the DDHM for collision detection and feedback purposes. Therefore an inverse transformation, from $U_V \rightarrow U_S$, can be executed to ensure values are scaled properly. The inverse transformation is achieved by generating the values according to equation (1). The inverse transformation can be used to regenerate the original elevation dataset from the DDHM and, as such, the DDHM can be used as the data source to describe the terrain surface in its original configuration for the rest of program execution.

$$\min(U_S) + V_{xz}(\max(U_S) - \min(U_S)) \tag{1}$$

3.4 Offset Generation

The primary task of this step is to calculate the complete set of displacement offsets for vertices that are subject to the compression forces of an object on the terrain. Deformation offset values are quickly and efficiently computed by rendering the depth buffer to a texture and then executing a fragment program to determine the appropriate offset values.

For the offset generation process, the viewing parameters of the camera are the same as during the initialization process. The inverse transformation is applied to the DDHM, and the resulting height map is rendered. During fragment assembly, the rendered elevation data is used to generate a copy of the DDHM by rendering to texture (Figure 2). Next, the object(s) that intersect the terrain surface are rendered (Figure 3). In the fragment shader, depth values are compared between the object and the contents of the copy of the DDHM. The object's depth information is only retained when the object is closer than the terrain, otherwise it is discarded. In this manner, the per-fragment comparison is used to determine the compression offset that the object imposes on the terrain. Upon completion of this first pass, an offset map is generated in video memory to store one elevation offset for each height in the DDHM.

3.5 Terrain Deformation

In the second render pass, the DDHM and the offset map are used to render the terrain, in its deformed state, to the screen. The algorithm uses the vertex shader to dynamically apply the deformation to the terrain.

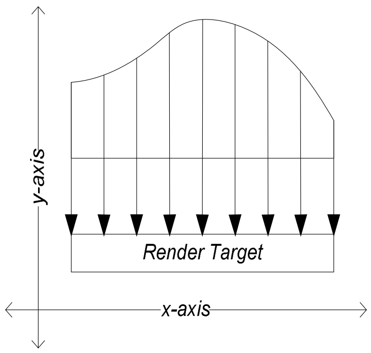


Fig. 2. The DDHM information is rendered into the render target

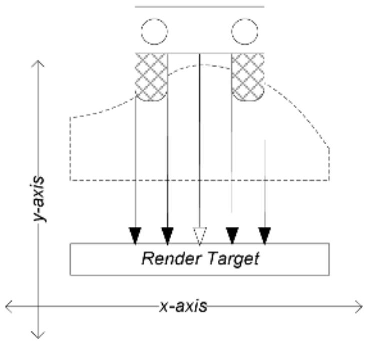


Fig. 3. The object depth information is rendered to the target when it is closer than the DDHM depth

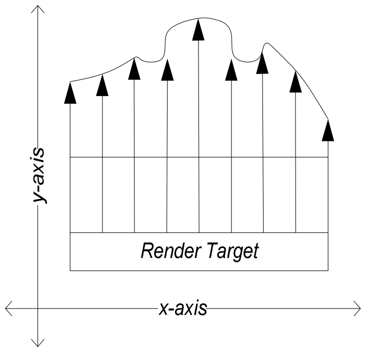


Fig. 4. The target buffer is used to repopulate the DDHM with the new elevation data

During this step, both the DDHM and the offset map are available in video memory and accessible as textures from the preceding render pass. Shader Model 3.0 offers vertex texture lookups, which can be used to achieve hardware-based displacement mapping. Using the vertex texture lookup functionality, each vertex of a rectilinear, planar grid can access the DDHM elevation value and the offset value through a common texture coordinate. The inverse transformation is applied to the DDHM value to determine the base height of the vertex. The offset is additively conjoined to the base height to produce the deformed height (Figure 4). Non-zero offset values indicate that a compression of the base height has occurred due to object imposition. The final height is prescribed to the vertex, which can be further processed; applying texturing and shading according to the systems requirements. In this manner, the terrain is deformed to realistically reflect soil compression.

4 Implementation and Experiments

One application for our method is visualizing the impact of a vehicle traversing soft terrain. To test this application, we have implemented a simple simulation using a ray-cast vehicle with physics-based subsystems. As the vehicle traverses the terrain, it leaves a trail of displaced vertices. The offset information is also used to modulate the base color of the terrain. The technique is visually effective as seen in figures 5, 6 and 7.

Collision detection and rigid body dynamics are performed using the Tokamak physics engine (http://www.tokamakphysics.com/). The simulation is written in C++ using OpenGL for rendering. The shaders are written in GLSL, a high level shader language for OpenGL. We use the frame buffer object (FBO) OpenGL extension to perform render to texture operations. Three floating point format textures each with a resolution of 1024x1024 are created. One texture is used to store the height map of the terrain, another is used to store the current depth profile of the vehicle, and the third is used to store the DDHM. For each pass of the algorithm, a different texture is bound to the color attachment of the FBO. After the third pass, the height map and vehicle depth profile will contain data that is used to compute the DDHM.

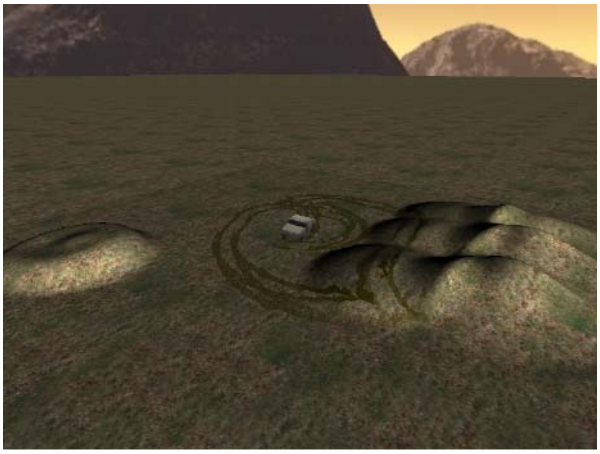


Fig. 5. The path the vehicle has traversed is clearly visible even across different elevations

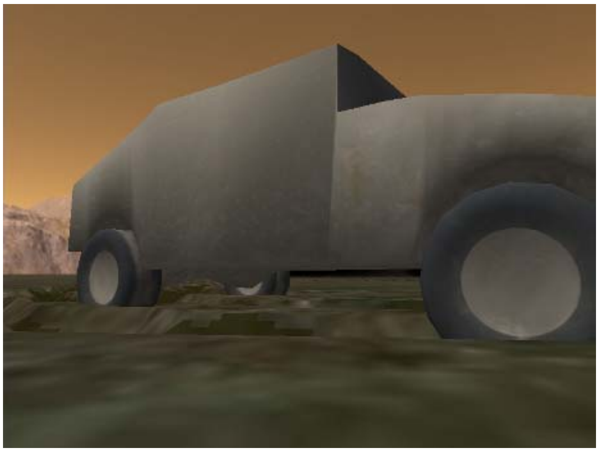


Fig. 6. Occlusion of the tires is possible because the terrain geometry is displaced

The DDHM is used by the vertex texture fetch operation in a vertex shader to displace the vertices of the terrain. Directly beneath the visual representation of the terrain is a lower density approximation of the terrain that is used by the physics engine to perform collision detection with the vehicle wheels. This also serves to limit the amount of displacement that the vehicle may exert on the terrain. During the final rendering pass of the terrain, a pixel shader uses the DDHM to modulate the color texture applied to the terrain, thus giving the impression of a muddy track left by the vehicle's tires.

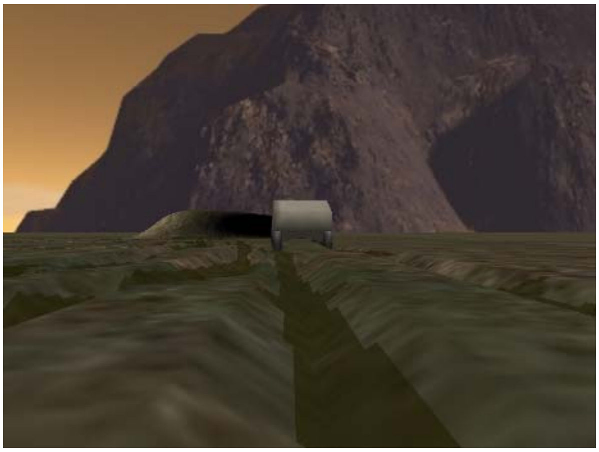


Fig. 7. Rutting caused by the vehicle's tires

The simulation runs at approximately 60 frames per second on a laptop computer with a Pentium M 1.6 GHz processor and GeForce 6800Go GPU, and at approximately 100 frames per second on a Pentium Xeon 2.0 GHz processor and

GeForce 7800 GTS GPU. Because frame buffer objects can both read from and write to textures, all operations are performed on the GPU thus eliminating costly transfers to and from the CPU. In order to ensure we are measuring the speed of computing the DDHM and displacing the vertices, physics and user input are disabled during our benchmarking trials. The full simulation with physics runs at approximately 40 and 70 frames per second on our two benchmarking machines respectively.

5 Analysis

Dynamic terrain visualization offers more contextual meaning to the scene and provides greater realism to the simulation thereby, improving the relevance of the simulation and the usefulness of the application. The method presented here offers fast, dynamic terrain deformation. Unlike previous methods, our algorithm takes advantage of several new features of the latest graphics hardware, such as vertex textures and frame buffer objects. As a result, our method achieves a high frame rate and is suitable for interactive 3D applications. Ground-vehicle simulation is an exemplary situation to use dynamic terrain, but the technique is flexible and allows for any rigid-body object to impose topographical changes.

Our method has some limitations. At the moment, our method does not simulate soil erosion. This means that it is best-suited for representing clay-like ground materials. In addition, like most previous methods, our current strategy assumes a limited terrain area, and thus is not appropriate for use in very large-scale terrains.

6 Conclusion and Future Work

We have presented a new algorithm for simulating dynamic, deformable terrain. Our algorithm achieves real-time performance by offloading most of the terrain manipulations to the GPU. To do this, we introduce the Dynamically-Displaced Height Map and the corresponding offset map, both of which are generated and maintained on the GPU. This algorithm takes advantage of some new features in the latest graphics hardware and shader technology, such as vertex textures and frame buffer objects.

Our algorithm is particularly useful for ground based visual simulation applications as well as 3D games. We have demonstrated the effectiveness of our algorithm by implementing a real-time driving simulation in which an off-road ground vehicle travels on soft terrain. Our current method is most suitable for simulating clay-like ground materials. In the future, we plan to extend our method to include granular materials by integrating an erosion mode. We also plan to extend our method to work with large-scale terrain.

References

1. Duchaineau, M., Wolinsky, M., Sigeti, D. E., Miller, M. C., Aldrich, C., and Mineev-Weinstein, M. B.: ROAMing terrain: real-time optimally adapting meshes, in Proceedings of the 8th IEEE Visualization Conference (1997)

2. Lindstrom, P., Koller, D., Ribarsky, W., Hodges, L. F., Faust, N., and Turner, G. A.: Real-time, continuous level of detail rendering of height fields, in Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH), (1996)
3. Lindstrom, P., and Pascucci, V.: Visualization of large terrains made easy, in Proceedings of the IEEE Visualization Conference (2001)
4. Losasso, F. and Hoppe, H.: Geometry clipmaps: terrain rendering using nested regular grids, ACM Transactions on Graphics, 23 (2004) 769-776
5. Hoppe, H.: Smooth view-dependent level-of-detail control and its application to terrain rendering, in Proceedings of the IEEE Visualization Conference (1998)
6. Li, X. and Moshell, J. M.: Modeling soil: realtime dynamic models for soil slippage and manipulation, in Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH) (1993)
7. Chanclou, B., Luciani, A., and Habibi, A.: Physical models of loss soils dynamically marked by a moving object, in Proceedings of the 9th IEEE Computer Animation Conference (1996)
8. Sumner, R. W., O'Brien, J. F., and Hodgins, J. K.: Animating sand, mud, and snow, Computer Graphics Forum, 18 (1999) 17-28
9. Onoue, K. and Nishita, T.: An interactive deformation system for granular material, Computer Graphics Forum, 24 (2005) 51-60
10. Ogren, A.: Continuous level-of-detail in real-time terrain rendering, Master's Thesis, University of Umea, Umea, Sweden (2000)

High-Resolution Video from Series of Still Photographs

Ge Jin and James K. Hahn

Department of Computer Science, The George Washington University,
Washington DC, 20052, USA
{jinge, hahn@gwu.edu}

Abstract. In this paper, we explored the problem of creating a high-resolution video from a series of still photographs. Instead of enhancing the resolution from the video stream, we consider the problem of generating a high-resolution video as an image synthesis problem. Using the continuous shot in the digital camera, we can get a series of still photographs at 2 to 3 frames pre second. The main challenge in our approach is to synthesize the in between frames from two consecutive still images. The image synthesis approach varies based on the scene motion and image characteristics. We have applied optical flow, image segmentation, image filtering and skeleton based image warping techniques to generate high-resolution video.

Keywords: Video Synthesis, Optical Flow, Image Segmentation.

1 Introduction

In spite of continuous development in digital video technology, making a movie-quality video is still beyond the limit of consumer-level digital camcorder. Compared with digital camcorder, the consumer-grade digital camera can capture images at much higher quality and resolution. Currently, more and more digital cameras are capable of taking continuous shot images. Continuous shot images can be considered as an intermediate stage between a single image and a video clip. These continuous images have a high-resolution feature of still image, while preserving the continuous motion cues in the video clip. Fast professional digital and sports camera can take continuous shot images at extremely high frame rates and these images can be directly converted to a video. However, for the consumer level digital camera, the frame rate of the continuous shot images is still limited to 2 to 3 frames per second. In this paper, we will concentrate on the problem of using low frame rate continuous shot images to synthesize high resolution video.

Fully automated video synthesis from any type of continuous images would be the ultimate goal. However, to prove the feasibility of making videos from continuous shot images, we have simplified the problem by limiting the types of scenes and allowing user interaction during the image segmentation stage. In this paper, we have limited our scenes to four different types of motions: static environment with moving camera, static camera with moving rigid object, static camera with character walking and static camera with moving trees. A general scene may consist of all these motions simultaneously, but we have tackled these problems separately.

In order to make a high resolution video from continuous shot images, we need to synthesize in between frames from two consecutive images. For the static environment with a moving camera, we used optical flow based image warping to generate in between images. For the rigid moving object, we used image segmentation and image morphing to synthesize in between frames. For the human walking scenes, we combined optical flow based and skeleton based image warping to produce human walking images. Finally, for the moving trees, we applied simple alpha blending followed by contrast enhancement filtering to generate the images of moving trees.

Most of algorithms described in this paper are fairly simple modifications of already known techniques. So, the main contribution of this paper is formulating the problem of high resolution video synthesis from still photographs, and proving the feasibility of our approach by performing experiments on four different types of moving scenes.

The rest of this paper is organized as follows: In section 2, we will look into some related works. The section 3 will provide detailed description of synthesizing in between images from different types of continuous shot images. We will show our experiment and result in section 4 and conclusion and future works in section 5.

2 Related Works

Dense optical flow method generates one to one pixel matching between two images. [1] calculated the optical flow using steepest descendent energy minimization method. Another well known optical flow algorithm is local motion optimization using the KLT method [2][3]. Further, the computation speed is increased by hierarchical searching [4] at different image scales. Recently, [5] describes a video matching method to align different video sources. They used robust estimation of flow field to interpolate and extrapolate at missing areas. We choose the KLT method for our optical flow calculation, for its efficiency and excellent result in quickly moving rigid objects. For our purpose, since the viewpoint or the motion is restricted by the two consecutive images, we do not need to recover the camera position and 3D information. In this context, our work is more related with feature based image morphing [6], skeleton based image warping [7] and view morphing works [8]. Commercial product like Apple Shake is using optical flow for re-timing, however, if the motion between two consecutive frames is large, the synthesized result presents motion-blur effect.

Image-based rendering focuses on generating new images and videos from given images. Some works used single image to generate a new image or an animated video. Tour into the picture [9] used a vanishing point to separate the image into spidery mesh regions and warped the image regions based on the camera movement. QuickTime VR [10] used panoramic image to generate camera panning and zooming in static environment. Stochastic motion texture is used to animate the movement of passive elements driven by the wind [11]. These works have produced convincing results. However, since there is only one image, the changes in view point or the animated motion is still limited to some small scale. We consider our work as a natural extension of video synthesis from a single image. Stop motion animation with image based motion blur is proposed to synthesize in between frames, where the image segmentation and optical flow techniques are used to synthesize motion blur effect [12]. Our proposed method differs from [12]'s work, where we are focused on synthesizing clear trajectory of the motion path. Motion magnification [13] is proposed to magnify the small motions in image sequence, however, for our case the motion in two consecutive images are relatively large and as a result, it is difficult to apply their method for our case.

In the image segmentation, graph cut based methods have produced good results both in still images and videos [14][15][16]. To extract clear alpha matte, Bayesian matting [17] and Poisson matting [18] approaches have been introduced to segment hairy objects. Another simple and efficient segmentation method is based on cellular automata [19]. For the video synthesis, the segmented image usually needs user refinement to remove the flickering artifact. So, for our case, we used the cellular automata based method to perform the initial segmentation and refine the result with user interaction.

3 Image Synthesis Using Continuous Shot Images

In order to synthesize a new image from two given images, we matched two consecutive images by calculating the optical flow and used the flow field to warp the source image to the destination image. For the static environment with moving camera, the optical flow with pseudo inverse image warping produces convincing results. However, for the scene with moving objects, only using optical flow could not generate good results. Some artifacts will occur at the boundaries of moving objects. So, we used the cellular automata based method to segment out the foreground layers and further refined it by user interaction. By using the optical flow at the foreground object layer, we morphed the image of a moving object to the in between frames. For the articulated figure such as walking human, image matching could not establish correct pixel mapping between walking legs. We have separated the human body into upper body and lower legs. For the upper body, we consider it as a rigid object. For the legs, we calculated the skeleton structure at each image, and used the skeleton based image warping to warp the legs. For the moving trees, we have applied simple alpha blending followed by contrast enhancement filtering to generate the in between frames.

3.1 Optical Flow Based Image Warping

Synthesis of in between frames for the camera moving at a static environment is image matching and warping problem. If the camera is panning at a fixed viewpoint, the problem becomes a panoramic image stitching and projective warping problem. For our case, we experimented with the camera moving along a certain path. Since we are only using one camera and not calculating the camera internal parameters and positions, the image matching between two adjacent frames is purely image based. We are dealing with the images that taken at a short time interval (about 0.5 second), the optical flow between two consecutive images will not be too large. Suppose $I(x, y)$ and $J(x, y)$ are two adjacent images. The optical flow vector $d = [d_x, d_y]$ maps the image point $u = [u_x, u_y]$ in image I to the image point $v = [v_x, v_y]$ in image J.

$$J(v_x, v_y) = I(u_x + d_x, u_y + d_y) \tag{1}$$

The optical flow vector d is calculated by minimizing the residual error at $u(u_x, u_y)$ with a certain window size.

$$\epsilon(d) = \epsilon(d_x, d_y) = \sum_{x=u_x-w_x}^{x=u_x+w_x} \sum_{y=u_y-w_y}^{y=u_y+w_y} (I(x, y) - J(x + d_x, y + d_y))^2 \tag{2}$$

The Pyramid Lucas Kanade optical flow algorithm calculates the flow vector at small image scale and propagates it into the next level as an initial guess. The error minimization is done by a gradient descendent method. In this paper, we did not propose a new optical flow algorithm. Instead, we used the pyramid LK optical flow library in OpenCV [20] to calculate the optical flow between two images.

Once we get the optical flow between two images, we can use the flow vector to warp one image to the other. Mathematically, it is a forward warping from image I to J using optical flow. However, the forward mapping has some problems with aliasing and holes. It can be done with two-pass method, but since the motion flow field is very smooth, we used pseudo inverse warping similar with [11]'s work. We used a linear factor f to smoothly change the flow vector from $[0, 0]$ to $[d_x, d_y]$. Since the frame rate of continuous shot images is about 2-3 frames per second, we need to generate about 10 images between two continuous shot images. So the linear factor increases from 0 to 1 with the interval of 0.1. The pseudo inverse warping function is described in equation (3).

$$I^*(x, y) = I(x - f * d_x, y - f * d_y) \tag{3}$$

The pseudo inverse warping works well on the scenes with a smooth vector field. However, when there is a moving object or the scene topology changes a lot, we could not simply warp the image using optical flow.

3.2 Movement of a Rigid Object

For the static camera with a moving object, we need to segment out the moving object first. To do this, we initially calculated the background image by averaging the full sequence of images. The background image was further refined by using a large threshold value (Summed RGB Color Difference) to disregard the foreground pixels and only use the remaining pixels to get an averaged background. To get more accurate background, this process is repeated by decreasing the threshold value. For our case, two steps of iteration with large and small threshold value have generated good results.

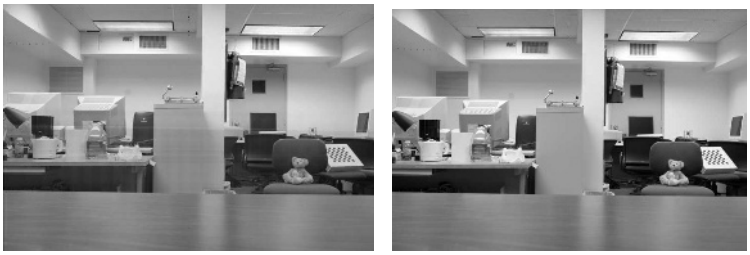


Fig. 1. Left: Background estimation with averaging, Right: Refined with large and small threshold

In the next step, we used simple RGB color threshold to segment out the foreground object. The result is further refined by eroding and dilating operation to remove background pixels and fill the holes inside foreground object. After this, we set the pixels at segmentation boundary as uncertain pixels.

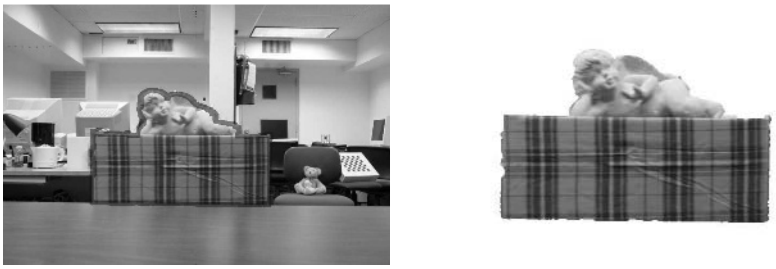


Fig. 2. Left: Segmentation with uncertain pixels marked in dark gray, Right: Segmentation result with Cellular Automata

Finally, we used the cellular automata based method to segment out the foreground object [19]. Each pixel is assigned with label, power, and its RGB color value. The label has three types: foreground, background and uncertain. The power of foreground and background pixels is set to 1 and the uncertain pixels are set to 0. At each iteration step, the neighbouring pixel with higher power and color similarity will try to change current pixel's label and power.

L_i:Label of Pixel i; P_i:Power of Pixel i(0 to 1)
RGB_i: RGB value of pixel i; $g(RGB_i - RGB_j) = 1 - \frac{\|RGB_i - RGB_j\|}{Max\|RGB\_Diff\|}$

```
for ∀ pixel j ∈ Neighbor of pixel i
    if $(g(RGB_j - RGB_i) * P_j^t > p_i^t$
        $L_i^{t+1} = L_j^t;$            $(t:\ iteration\ step)$
        $P_i^{t+1} = g(RGB_j - RGB_i) * P_j^t;$
    endif
end for
```

The automatic segmentation result with cellular automata is quite good. In order to remove the flicking effect, we improved the segmentation result by moving the boundary lines as described in [14].After segmentation, we calculated the optical flow of the moving object using the method described in section 1. The segmented images usually do not perfectly match and there will be small variations in camera focus and lighting conditions. Because of this, simply warping the source image to the destination image will cause flickering effect. Thus, we used image morphing to generate the in between frames. First, we warped the destination image to the source image using optical flow and got a warped source image J^*. All the pixel value in image J^* comes from the destination image. A new image is generated by linearly blending source image I and warped source image J^*. The morphing equation is described in equation 4.

$$I^*(x,y) = (1-f) * I(x - f * d_x, y - f * d_y) + f * J^*(x - f * d_x, y - f * d_y) \quad (4)$$

For the reflections and shadows, if we use image warping at the reflection part, it will make some changes to the texture of the background scene. So, we used the difference image and warp the difference image using optical flow. The difference image does not contain good features, and this will cause some abrupt changes in the synthesized reflection image. So we used the low pass filtering to smooth the warped reflection images.

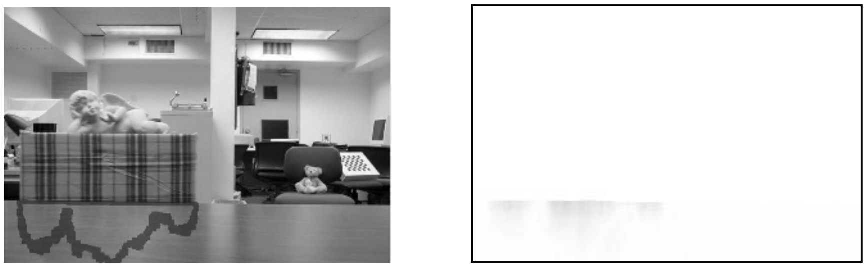


Fig. 3. Left: Reflection Image. Right: Difference Image.

3.3 Human Walking

For the walking scenes, we have separated the human body into upper body and lower legs. For the upper body, we consider it as a rigid moving object and the in between images are generated using the method described in section 2. For the legs, we calculated the skeleton structure at each image, and used the skeleton based image warping to warp the legs from one image to the other. At

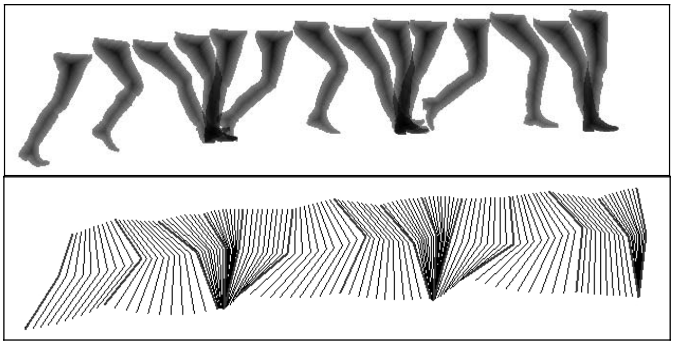


Fig. 4. Upper image: Lower legs in image sequence; Lower image: In between skeleton calculation with forward kinematics

each inner pixel inside the segmented matte, we calculated the shortest pixel distance to the matte boundary. The skeleton structure is easily calculated by searching for the local maxima and the hough transform. The in between skeleton structure is calculated using a simple forward kinematics method. We used the feature based image morphing described in [6] to warp the leg images. At the boundaries of upper body and lower legs, there will be some discontinuity because of different image warping methods. So at these boundaries, we calculated the pixel displacement by linearly blending the displacement from optical flow based warping and skeleton based warping. The left leg is warped in a similar way. One

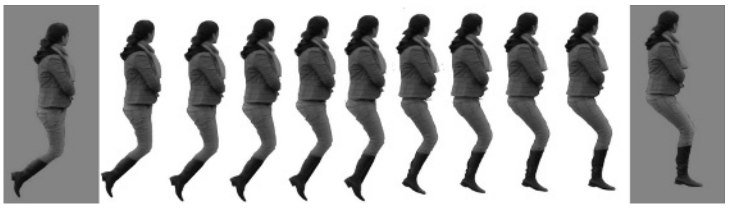


Fig. 5. The result of combining the upper body and lower leg

problem with left leg is: some of the left legs are heavily occluded by the right legs. For those heavily occluded left legs, we manually painted those left legs using nearby left leg images.

3.4 Trees

The tree movement has stochastic motion feature, so, it is very difficult to use image matching based method to generate the in between frames. We have applied a simple alpha blending followed by debluring operation to generate the in between frames. If we play the alpha blended image sequence, it will look like highly motion blured video. So we used the contrast enhancing filter to decrease the bluring effect. However, the contrast enhancing operation on original images will make the original continuous shot images much sharper than the alpha blended images. So we used simple method to blur the original images. I_0 is the original continuous shot image, I_{-1} and I_1 are the continuous shot images before and after I_0. The new alpha blended image is generated using equation (5).

$$I_0^*(x,y) = 0.1 * I_{-1}(x,y) + 0.8 * I_0(x,y) + 0.1 * I_1(x,y) \tag{5}$$

With this simple approach, we can generate a slightly motion blured video of tree movement.

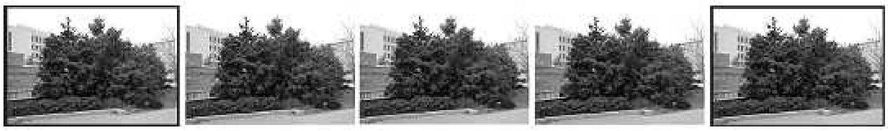


Fig. 6. The result of tree motion synthesis

4 Experiment and Result

We used Canon 20D digital camera, with remote switch to capture continuous shot images. We can capture about 300 continuous images without interruption. Since the frame rate is 2-3 frames per second, it can be thought as a 3-minute video clip. The camera shutter speed needs to be fast. For indoor scenes, the shutter speed is set to 1/60–1/40, and the ISO is set to 1600. For outdoor scenes, the shutter speed is set to 1/200–1/125, and the ISO is set to 100-200. We used the smallest image size (1728 X 1152) for capturing continuous images. For our experiment, we down sampled the image to 1440 X 960. We used workstation with Intel 3.2Ghz Xeon Processor and 2GB memory for our experiment. For the programming, we used Visual C++ with OpenCV library and Matlab6.5.

In our video synthesis, the main speed bottleneck is the optical flow calculation. Since we are using the pseudo inverse image warping, the window size in optical flow calculation is set to 15–30. For the camera moving at a static environment, the window size is set to 15. For the moving objects and human walking, since the motion is large, we set the window size to 30. The optical flow computation time is about 180 seconds at image size 1440 X 960 with window size 15. For the window size 30, the computation time is about 720 seconds. Since the optical flow computation takes too much time, we have experimented using the optical flow at a smaller image size, and linearly interpolated to get the enlarged optical flow at a bigger image size. The optical flow computation at 720 X 480 with window size 15 is about 50 seconds. The image synthesis time using inverse warping is less than one second per image. For the camera moving at a static environment, the warped image has some holes at the image boundary. We simply cropped the whole sequence of synthesized images to generate a high resolution video. The cropped image is 1080 X 720, and it is still much bigger than the images from digital camcorder. For our human walking video, since we only used simple forward kinematics based approach, the resultant human walking presents some moon walking effect.

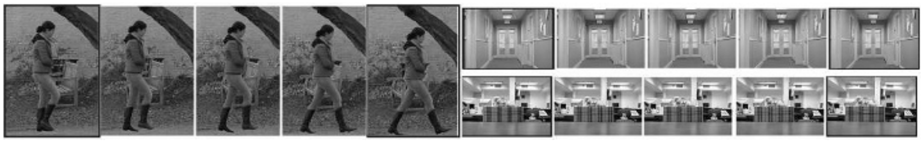


Fig. 7. The result of proposed method

5 Conclusions and Future Works

In this paper, we introduced a new way to synthesize a high resolution video. Instead of enhancing the resolution from video stream, we used continuous shot still images to synthesize the video. In image synthesis using adjacent frames, we have found that the approaches need to be varied based on the motion in the scene. We have simplified the problem by limiting the types of scenes. For the static environment, we used optical flow with pseudo inverse image warping. For the rigid moving object, we used segmentation and optical flow to animate the rigid moving objects. For the reflected image, we used the difference image for image warping and applied low pass filtering to smooth the result. For the human walking, we combined optical flow based image warping and skeleton based image warping to generate the human walking video. Finally, we used simple alpha blending and contrast enhancement filtering to generate the video of tree movement. With our approach, it is possible to make a movie quality video using consumer level digital cameras. Even in the future, as long as the resolution of digital cameras is higher than the digital camcoders, our approach is an alternative way to overcome the resolution limit of digital video devices. For the future work, we want to implement optical flow algorithm on GPU. Many video based rendering approaches are depending on optical flow calculation, but the optical flow computation is too slow for the interactive applications. We will experiment on different types of scenes, for example, driving scenes, moving camera with moving objects, waving water and so on. We also think about using a carefully calibrated video and camera pair. The continuous shot image from the camera is going to be warped based on the optical flow from the video image sequence.

References

1. Horn B., Schunck B.: Determining Optical Flow. In Artificial Intelligence, Vol. 17: (August 1981),185–203.
2. Lucas B., Kanade T.: An iterative image registration technique with an application to stereo vision. In Proc. of International Joint Conference on Artificial Intelligence, 1981. 674–679.
3. Shi J., Tomasi C.: Good features to track. In IEEE CVPR'1994, 593–600.
4. Bouguet, J. Y.: Pyramidal Implementation of the Lucas Kanade Feature Tracker. Intel Corporation, Microprocessor Research Labs, 2000,
5. Sand P., Teller S.: Video matching. In ACM Transactions on Graphics, Vol. 23(3):(2004), 592–599
6. Beier T., Neely S.: Feature-based image metamorphosis. In Proc. ACM SIGGRAPH, (July 1992), 35–42.
7. Wolberg G.: Skeleton Based Image Warping. In Visual Computer, Vol .5(1): (1989) 95–108
8. Seitz S.M., Dyer C. R.: View morphing. In Proc. ACM SIGGRAPH'1996, 21–30.
9. Horry Y., Anjoy K., Arai K.: Tour into the picture: using a spidery mesh interface to make animation from a single image. In Proc. ACM SIGGRAPH, (Aug. 1997), 225–232.
10. Chen S. E.: Quicktime VR - an image-based approach to virtual environment navigation. In Proc. of ACM SIGGRAPH'95 (1995), 29–38.
11. Chuang Y.Y., Goldman D.B., Zheng K.C., Curless B., Salesin D.H., Szeliski R.: Animating pictures with stochastic motion textures. In ACM Transactions on Graphics, Vol. 24(3), 2005. 853–860.
12. Brostow, G. J. and Essa, I.: Image-based motion blur for stop motion animation. In Proc. of ACM SIGGRAPH '01, (2001) 561–566
13. Liu, C., Torralba, A., Freeman, W. T., Durand, F., and Adelson, E. H.: Motion magnification. In ACM Trans. Graphics, Vol. 24(3): (2005), 519–526
14. Li Y., Sun J., Tang C.K., Shum H.Y.: Lazy snapping. In ACM Transactions on Graphics, Vol. 23(3): (2004), 303–308.
15. Rother C., Kolmogorov V., Blake A.: "GrabCut": interactive foreground extraction using iterated graph cuts. In ACM Trans. on Graphics, Vol. 23(3): (2004),309–314.
16. Li Y., Sun J., Shum H.Y.: Video object cut and paste. In ACM Transactions on Graphics, Vol. 24(3): (2005), 595–600
17. Chuang Y.Y., Agrawala M, Curless B., Salesin D.H., Szeliski R: Video matting of complex scenes. In Proceedings of ACM SIGGRAPH, (2002), 243–248
18. Sun J., Jia J., Tang C.K., Shum H.Y.: Poisson matting. In ACM Transactions on Graphics, Vol. 23(3): (2004), 315–321.
19. Vezhnevets V., Konouchine V.: "Grow-Cut" - Interactive Multi-Label N-D Image Segmentation. In Int. conf. on the Computer Graphics and Vision (Graphicon 2005), 150–156.
20. OpenCV: OpenCV: The Open Computer Vision Library. Sourceforge.net/projects/opencvlibrary.

Author Index