José Braz
Alpesh Ranchordas
Hélder J. Araújo
João Madeiras Pereira (Eds.)

Communications in Computer and Information Science 21

Computer Vision and Computer Graphics

Theory and Applications

International Conference VISIGRAPP 2007
Barcelona, Spain, March 2007
Revised Selected Papers

Springer

Communications
in Computer and Information Science 21

José Braz Alpesh Ranchordas
Hélder J. Araújo João Madeiras Pereira (Eds.)

Computer Vision and Computer Graphics

Theory and Applications

International Conference VISIGRAPP 2007
Barcelona, Spain, March 8-11, 2007
Revised Selected Papers

Springer

Volume Editors

José Braz
Departamento de Sistemas e Informatica
Escola Superior de Tecnologia do IPS
2910 Setúbal, Portugal
E-mail: jbraz@est.ips.pt

Alpesh Ranchordas
INSTICC
2910 Setúbal, Portugal
E-mail: alpesh@visapp.org

Hélder J. Araújo
University of Coimbra
Institute for Systems and Robotics
Polo II, 3030-290 Coimbra, Portugal
E-mail: helder@isr.uc.pt

João Madeiras Pereira
IST/INESC-ID
1000-029 Lisboa, Portugal
E-mail: jap@inesc.pt

Library of Congress Control Number: 2008940337

CR Subject Classification (1998): I.2.10, I.4, I.5.4, I.6

ISSN 1865-0929
ISBN-10 3-540-89681-3 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-89681-4 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12562732 06/3180 5 4 3 2 1 0

Preface

This book includes selected papers from VISIGRAPP 2007, the Joint Conference on Computer Vision and Computer Graphics, comprising two component conferences, namely, the International Conference on Computer Vision Theory and Applications (VISAPP) and the International Conference on Computer Graphics Theory and Applications (GRAPP), held in Barcelona, Spain, during March 8–11, 2007.

We received quite a high number of paper submissions: 382 in total for both conferences. We had contributions from more than 50 countries in all five continents. This confirms the success and global dimension of these jointly organized conferences. After a rigorous double-blind evaluation method, a total of 78 submissions were accepted as full papers. From those, 18 got selected for inclusion in this book. To ensure the scientific quality of the contributions, these were selected from papers that were evaluated with the highest scores by the VISIGRAPP Program Committee members and then they were extended and revised by the authors. Special thanks go to all contributors and referees, without whom this book would not have been possible.

VISIGRAPP 2007 included four invited keynote lectures, presented by internationally recognized researchers. The presentations represented an important contribution to increasing the overall quality of the conference. We would like to express our appreciation to all invited keynote speakers, in alphabetical order: Jake K. Aggarwal (The University of Texas at Austin/USA), André Gagalowicz (INRIA/France), Wolfgang Heidrich (University of British Columbia/Canada), Mel Slater (Universitat Politècnica de Catalunya/Spain).

We wish to thank all those who supported and helped to organize the conference. First and foremost we would like to acknowledge the collaboration from Eurographics and CVC - Computer Vision Center. Moreover, on behalf of the conference Organizing Committee, we would like to thank the authors, whose work mostly contributed to a very successful conference, and to the members of the Program Committee, whose expertise and diligence were instrumental to the quality of the final contributions. We also wish to thank all the members of the Organizing Committee whose work and commitment were invaluable. Last but not least, we would like to thank Springer for their collaboration in getting this book to print.

July 2008

José Braz
AlpeshKumar Ranchordas
Helder Araújo
João Madeiras Pereira

Conference Committee

Conference Co-chairs

José Braz	Polytechnic Institute of Setúbal, Portugal (GRAPP)
AlpeshKumar Ranchordas	INSTICC, Portugal (VISAPP)

Program Co-chairs

Hélder J. Araújo	Universidade de Coimbra, Portugal (VISAPP)
João Madeiras Pereira	IST/INESC-ID, Portugal (GRAPP)
Pere-Pau Vázquez	Universitat Politècnica de Catalunya, Spain (GRAPP)
Jordi Vitrià	Universitat Autònoma de Barcelona, Spain (VISAPP)

Organizing Committee

Paulo Brito	INSTICC, Portugal
Marina Carvalho	INSTICC, Portugal
Helder Coelhas	INSTICC, Portugal
Andreia Costa	INSTICC, Portugal
Vítor Duarte	INSTICC, Portugal
Bruno Encarnação	INSTICC, Portugal
Luís Marques	INSTICC, Portugal
Vitor Pedrosa	INSTICC, Portugal
Mónica Saramago	INSTICC, Portugal

GRAPP Program Committee

Francisco Abad, Spain
Sergey Ablameyko, Belarus
Marco Agus, Italy
Trémeau Alain, France
Daniel Aliaga, USA
Carlos Ureña Almagro, Spain
Marco Attene, Italy
Dolors Ayala, Spain
Sigal Ar, Israel
Sergei Azernikov, USA
Jacob Barhak, USA
William Baxter, Japan
Rafael Bidarra, The Netherlands
Jiri Bittner, Czech Republic
Manfred Bogen, Germany
Kadi Bouatouch, France
Ronan Boulic, Switzerland
Willem F. Bronsvoort, The Netherlands
Pere Brunet, Spain
Guido Brunnett, Germany

Sam Buss, USA
Patrick Callet, France
Emilio Camahort, Spain
Pedro Cano, Spain
Juan Carlos Torres Cantero, Spain
Maria Beatriz Carmo, Portugal
Leocadio González Casado, Spain
Teresa Chambel, Portugal
Chun-Fa Chang, Taiwan
Norishige Chiba, Japan
Eng-Wee Chionh, Singapore
Hwan-Gue Cho, Korea
Min-Hyung Choi, USA
Miguel Chover, Spain
Yiorgos Chrysanthou, Cyprus
Ana Paula Cláudio, Portugal
Sabine Coquillart, France
Nuno Correia, Portugal
António Cardoso Costa, Portugal
Balázs Csébfalvi, Hungary
Carsten Dachsbacher, France
Hervé Delingette, France
John Dingliana, Ireland
Jean-Michel Dischler, France
Sasa Divjak, Slovenia
Stéphane Donikian, France
David Duce, UK
Roman Durikovic, Japan
James Edge, UK
Francisco R. Feito, Spain
Petr Felkel, Czech Republic
Inmaculada Garcia Fernandez, Spain
Fernando Nunes Ferreira, Portugal
Luiz Henrique de Figueiredo, Brazil
Pablo Figueroa, Colombia
Anath Fischer, Israel
Eugene Fiume, Canada
Julian Flores, Spain
Leila De Floriani, Italy
Martin Fuchs, Germany
Ioannis Fudos, Greece
Tadahiro Fujimoto, Japan
Aphrodite Galata, UK
Manuel Gamito, UK
Marina Gavrilova, Canada
Miguel Gea, Spain
Michael Gleicher, USA
Mashhuda Glencross, UK
Enrico Gobbetti, Italy
Michael Goesele, USA
Abel Gomes, Portugal
Eduard Gröller, Austria
Tom Gross, Germany
Alain Grumbach, France
Stefan Gumhold, Germany
Igor Guskov, USA
Mario Gutierrez, Mexico
Peter Hall, UK
Helwig Hauser, Austria
Vlastimil Havran, Czech Republic
José Tiberio Hernandez, Colombia
Adrian Hilton, UK
Nancy Hitschfeld, Chile
Christoph M. Hoffmann, USA
Toby Howard, UK
Zhiyong Huang, Singapore
Roger Hubbold, UK
Insung Ihm, Korea
Andres Iglesias, Spain
Patricio Inostroza, Chile
Jiri Janacek, Czech Republic
Frederik Jansen, The Netherlands
Jie Tang, China
Robert Joan-Arnyo, Spain
Andrew Johnson, USA
Chris Joslin, Canada
Marcelo Kallmann, USA
Arie Kaufman, USA
Jan Kautz, UK
HyungSeok Kim, Korea
Young J. Kim, Korea
Yoshifumi Kitamura, Japan
Stanislav Klimenko, Russia
Josef Kohout, Czech Republic
Ivana Kolingerova, Czech Republic
David Laidlaw, USA
Won-Sook Lee, Canada
Miguel Leitão, Portugal
Heinz U. Lemke, Germany
Hendrik Lensch, Germany
Suresh Lodha, USA
Celine Loscos, UK
Steve Maddock, UK
Joaquim Madeira, Portugal

Nadia Magnenat-Thalmann, Switzerland
Marcus Magnor, Germany
Stephen Mann, Canada
Michael Manzke, Ireland
Michael McCool, Canada
Tom Mertens, Belgium
Dominique Michelucci, France
Laurent Moccozet, Switzerland
Ramon Molla, Spain
Torsten Möller, Canada
Claudio Montani, Italy
Matthias Mueller-Fischer, Switzerland
Franck Multon, France
Ken Museth, Sweden
Isabel Navazo, Spain
Laszlo Neumann, Spain
Gennadiy Nikishkov, Japan
Alexander Nischelwitzer, Austria
Marc Olano, USA
Victor Ostromoukhov, Canada
Miguel A. Otaduy, Switzerland
Zhigeng Pan, China
Georgios Papaioannou, Greece
Alexander Pasko, Japan
Nuria Pelechano, USA
João Pereira, Portugal
Bernard Péroche, France
Steve Pettifer, UK
Dimitri Plemenos, France
Voicu Popescu, USA
Anna Puig, Spain
Enrico Puppo, Italy
Werner Purgathofer, Austria
Ari Rappoport, Israel
Stephane Redon, France
Maria-Cecilia Rivara, Chile
Przemyslaw Rokita, Poland
Daniela Romano, UK
Bodo Rosenhahn, Germany
Manuel Próspero dos Santos, Portugal
Muhammad Sarfraz, Saudi Arabia
Francis Schmitt, France
Rafael J. Segura, Spain
Ariel Shamir, Israel
Ilan Shimshoni, Israel
Peter Shirley, USA
Deborah Silver, USA
A. Augusto Sousa, Portugal
Alexei Sourin, Singapore
Oliver Staadt, USA
Tapio Takala, Finland
Ayellet Tal, Israel
Tiow Seng Tan, Singapore
José Carlos Teixeira, Portugal
Matthias Teschner, Germany
Daniel Thalmann, Switzerland
Holger Theisel, Germany
Christian Theobalt, Germany
Gui Yun Tian, UK
Walid Tizani, UK
Anna Ursyn, USA
Francisco Velasco, Spain
Luiz Velho, Brazil
Frederic Vexo, Switzerland
Max Viergever, The Netherlands
Anna Vilanova , The Netherlands
Ivan Viola, Norway
Wenping Wang, China
Andreas Weber, Germany
Daniel Weiskopf, Canada
Gerold Wesche, Germany
Alexander Wilkie, Austria
Michael Wimmer, Austria
Burkhard Wuensche, New Zealand
Gabriel Zachmann, Germany
Jian J. Zhang, UK
Jianmin Zheng, Singapore
Alan Zundel, USA

VISAPP Program Committee

Henrik Aanæs, Denmark
Wael Abd-Almageed, USA
Samer M. Abdallah, Lebanon
Peggy Agouris, USA
Selim Aksoy, Turkey
Ernesto Andrade, UK
Helder Araújo, Portugal
Mohammed Bennamoun, Australia

Alexandre Bernardino, Portugal
Prabir Bhattacharya, Canada
S.A. Billings, UK
Nicolas Perez de la Blanca, Spain
Jacques Blanc-Talon, France
Isabelle Bloch, France
Alain Boucher, Vietnam
Pierre Boulanger, Canada
Salah Bourennane, France
Thomas Breuel, Germany
Christopher Brown, USA
Hans du Buf, Portugal
Rob Byrd, USA
Jens Michael Carstensen, Denmark
John Carter, UK
Roberto Cesar Jr., Brazil
Andrea Cavallaro, UK
Rama Chellappa, USA
Sheng Yong Chen, Hong Kong
Yung-Fu Chen, Taiwan
Hocine Cherifi, France
Ronald Chung, Hong Kong
James Clark, Canada
Isaac Cohen, USA
Fabio Cuzzolin, USA
Kostas Daniilidis, USA
Roy Davies, UK
Chris Debrunner, USA
Joachim Denzler, Germany
Jorge Dias, Portugal
John Dingliana, Ireland
Zoran Duric, USA
Laurent Duval, France
Aly Farag, USA
Marco Ferretti, Italy
Mário Figueiredo, Portugal
Robert Fisher, UK
Jan Flusser, Czech Republic
Alejandro Frangi, Spain
Donald Fraser, Australia
Sidharta Gautama, Belgium
Daniel Giusto, Italy
Vicente Grau, UK
Michael Greenspan, Canada
Igor Gurevich, Russia
Jiro Gyoba, Japan
Ghassan Hamarneh, Canada
Dan Witzner Hansen, Denmark
Allen Hanson, USA
Thomas Henderson, USA
Ellen Hildreth, USA
Joachim Hornegger, Germany
Hsi-Chin Hsin, Taiwan
Benoit Huet, France
Mark Huiskes, The Netherlands
John Illingworth, UK
Luca Iocchi, Italy
Michael Jenkin, Canada
Tianzi Jiang, China
Xiaoyi Jiang, Germany
Frédéric Jurie, France
George Kamberov, USA
Gerda Kamberova, USA
Bill Kapralos, Canada
Andrzej Kasinski, Poland
Ullrich Koethe, Germany
Esther Koller-Meier, Switzerland
Constantine Kotropoulos, Greece
Murat Kunt, Switzerland
Kyoung Mu Lee, Korea
Bastian Leibe, Switzerland
Ales Leonardis, Slovenia
Michael Lew, The Netherlands
Baoxin Li, USA
Ching-Chung Li, USA
Peihua Li, China
XueLong Li, UK
Jianming Liang, USA
Jenn-Jier James Lien, Taiwan
Angeles López, Spain
Rastislav Lukac, Canada
Ezio Malis, France
Geovanni Martinez, Costa Rica
Stephen Maybank, UK
Brendan McCane, New Zealand
Gerard Medioni, USA
Mahmoud Melkemi, France
J.P. Mellor, USA
Emanuele Menegatti, Italy
Ajmal Mian, Australia
Max Mignotte, Canada
Majid Mirmehdi, UK
Ali Mohammad-Djafari, France
Stefan Müller-Schneiders, Germany

Vittorio Murino, Italy
Hammadi Nait-Charif, UK
Bernd Neumann, Germany
Shawn Newsam, USA
Heinrich Niemann, Germany
Mark Nixon, UK
Lucas Paletta, Austria
Jussi Parkkinen, Finland
Arthur Pece, Denmark
Justus Piater, Belgium
Ioannis Pitas, Greece
Filiberto Pla, Spain
Dan Popescu, Australia
David Pycock, UK
Gang Qian, USA
Petia Radeva, Spain
Bogdan Raducanu, Spain
Paolo Remagnino, UK
Eraldo Ribeiro, USA
Edward Riseman, USA
Ovidio Salvetti, Italy
Muhammad Sarfraz, Saudi Arabia
Gerald Schaefer, UK
Li Shen, USA
Franc Solina, Slovenia
Domenico Sorrenti, Italy
José Martínez Sotoca, Spain
Peter Sturm, France
Jianbo Su, China
Sriram Subramanian, Canada
Shamik Sural, India
Yung-Nien Sun, Taiwan
Eric Sung, Singapore
Qi Tian, USA
Sinisa Todorovic, USA
John Tsotsos, Canada
Peter Veelaert, Belgium
Konstantinos Veropoulos, USA
Dimitri Van De Ville, Switzerland
Sven Wachsmuth, Germany
Ellen Walker, USA
Frank Wallhoff, Germany
Song Wang, USA
Uian-Kai Wang, Taiwan
Yongmei Michelle Wang, USA
Jonathan Wu, Canada
Shuicheng Yan, USA
Faguo Yang, USA
June-Ho Yi, USA
Dong Xu, USA
John Zelek, Canada
Ying Zheng, UK
Zhigang Zhu, USA
Djemel Ziou, Canada

GRAPP Auxiliary Reviewers

Edilson de Aguiar, Germany
Naveed Ahmed, Germany
Jean-Paul Balabanian, Norway
Raphael Bürger, Austria
Juergen Gall, Germany
Ralf Habel, Austria
Nils Hasler, Germany
Stephan Mantler, Austria
Oliver Mattausch, Austria
Peter Rautek, Austria
Daniel Scherzer, Austria
Robert Tobler, Austria
Zsolt Tóth, Slovakia
Erald Vuçini, Austria
Hongchuan Yu, UK

VISAPP Auxiliary Reviewers

Ksenia Shubina, Canada
Neil Bruce, Canada
Juan Dai, Hong Kong
Jacob Gryn, Canada
Erich Leung, Canada
Davide Moroni, Italy
Yuan Yuan, UK
Barbara Zitova, Czech Republic

Invited Speakers

André Gagalowicz	INRIA Rocquencourt, France
Mel Slater	Universitat Politècnica de Catalunya, Spain
Jake K. Aggarwal	The University of Texas at Austin, USA
Wolfgang Heidrich	University of British Columbia, Canada

Table of Contents

Part II: Image Analysis

Part III: Image Understanding

Part IV: Motion, Tracking and Stereo Vision

Computer Graphics Theory and Applications

Part I
Geometry and Modeling

Implicit Surface Reconstruction with Radial Basis Functions

Jun Yang[1], Zhengning Wang[2], Changqian Zhu[2], and Qiang Peng[2]

[1] School of Mechanical & Electrical Engineering Lanzhou Jiaotong University, Lanzhou, Gansu 730070, China
[2] School of Information Science & Technology Southwest Jiaotong University, Chengdu, Sichuan 610031, China
yangj@mail.lzjtu.cn, {znwang,cqzhu,pqiang}@home.swjtu.edu.cn

Abstract. This paper addresses the problem of reconstructing implicit function from point clouds with noise and outliers acquired with 3D scanners. We introduce a filtering operator based on mean shift scheme, which shift each point to local maximum of kernel density function, resulting in suppression of noise with different amplitudes and removal of outliers. The "clean" data points are then divided into subdomains using an adaptive octree subdivision method, and a local radial basis function is constructed at each octree leaf cell. Finally, we blend these local shape functions together with partition of unity to approximate the entire global domain. Numerical experiments demonstrate robust and high quality performance of the proposed method in processing a great variety of 3D reconstruction from point clouds containing noise and outliers.

Keywords: Filtering, space subdivision, radial basis function, partition of unity.

1 Introduction

The interest for point-based surface has grown significantly in recent years in computer graphics community due to the development of 3D scanning technologies, or the riddance of connectivity management that greatly simplifies many algorithms and data structures. Implicit surfaces are an elegant representation to reconstruct 3D surfaces from point clouds without explicitly having to account for topology issues. However, when the point sets data generated from range scanners (or laser scanners) contain large noise, especially outliers, some established methods often fail to reconstruct surfaces or real objects.

There are two major classes of surface representations in computer graphics: parametric surfaces and implicit surfaces. A parametric surface [1, 2] is usually given by a function $f(s, t)$ that maps some 2-dimensional (maybe non-planar) parameter domain Ω into 3-space while an implicit surface typically comes as the zero-level isosurface of a 3-dimensional scalar field $f(x, y, z)$. Implicit surface models are popular since they can describe complex shapes with capabilities for surface and volume modeling and complex editing operations are easy to perform on such models. Moving least square (MLS) [3-6] and radial basis function (RBF) [7-15] are two popular 3D implicit surface reconstruction methods.

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 5–12, 2008.

Recently, RBF attracts more attention in surface reconstruction. It is identified as one of most accurate and stable methods to solve scattered data interpolation problems. Using this technique, an implicit surface is constructed by calculating the weights of a set of radial basis functions such they interpolate the given data points. From the pioneering work [7, 8] to recent researches, such as compactly-supported RBF [9, 10], fast RBF [11-13] and multi-scale RBF [14, 15], the established algorithms can generate more and more faithful models of real objects in last twenty years, unfortunately, most of them are not feasible for the approximations of unorganized point clouds containing noise and outliers.

In this paper, we describe an implicit surface reconstruction algorithm for noise scattered point clouds with outliers. First, we define a smooth probability density kernel function reflecting the probability that a point $\mathbf{p}$ is a point on the surface S sampled by a noisy point cloud. A filtering procedure based on mean shift is used to move the points along the gradient of the kernel functions to the maximum probability positions. Second, we reconstruct a surface representation of "clean" point sets implicitly based on a combination of two well-known methods, RBF and partition of unity (PoU). The filtered domain of discrete points is divided into many subdomians by an adaptively error-controlled octree subdivision, on which local shape functions are constructed by RBFs. We blend local solutions together using a weighting sum of local subdomains. As you will see, our algorithm is robust and high quality.

2 Filtering

2.1 Covariance Analysis

Before introducing our surface reconstruction algorithm, we describe how to perform eigenvalue decomposition of the covariance matrix based on the theory of principal component analysis (PCA) [24], through which the least-square fitting plane is defined to estimate the kernel-based density function.

Given the set of input points $\Omega = \{\mathbf{p}_i\}_{i\in[1,L]}$, $\mathbf{p}_i \in \mathbb{R}^3$, the weighted covariance matrix $\mathbf{C}$ for a sample point $\mathbf{p}_i \in \Omega$ is determined by

$$\mathbf{C} = \sum_{j=1}^{L} \left(\mathbf{p}_j - \bar{\mathbf{p}}_i\right)\left(\mathbf{p}_j - \bar{\mathbf{p}}_i\right)^{\mathrm{T}} \cdot \Psi\left(\left\|\mathbf{p}_j - \mathbf{p}_i\right\| / h\right), \tag{1}$$

where $\bar{\mathbf{p}}_i$ is the centroid of the neighborhood of $\mathbf{p}_i$, Ψ is a monotonically decreasing weight function, and h is the adaptive kernel size for the spatial sampling density. Consider the eigenvector problem

$$\mathbf{C} \cdot \mathbf{e}_l = \lambda_l \cdot \mathbf{e}_l . \tag{2}$$

Since $\mathbf{C}$ is symmetric and positive semi-define, all eigenvalues λ_l are real-valued and the eigenvectors $\mathbf{e}_l$ form an orthogonal frame, corresponding to the principal components of the local neighborhood.

Assuming $\lambda_0 \le \lambda_1 \le \lambda_2$, it follows that the least square fitting plane H($\mathbf{p}$): $\left(\mathbf{p} - \bar{\mathbf{p}}_i\right) \cdot \mathbf{e}_0 = 0$ through $\bar{\mathbf{p}}_i$ minimizes the sum of squared distances to the neighbors of $\mathbf{p}_i$. Thus $\mathbf{e}_0$ approximates the surface normal $\mathbf{n}_i$ at $\mathbf{p}_i$, i.e., $\mathbf{n}_i = \mathbf{e}_0$. In other words, $\mathbf{e}_1$ and $\mathbf{e}_2$ span the tangent plane at $\mathbf{p}_i$.

2.2 Mean Shift Filtering

Mean shift [16, 17] is one of the robust iterative algorithms in statistics. Using this algorithm, the samples are shifted to the most likely positions which are local maxima of kernel density function. It has been applied in many fields of image processing and visualization, such as tracing, image smoothing and filtering.

In this paper, we use a nonparametric kernel density estimation scheme to estimate an unknown density function $g(\mathbf{p})$ of input data. A smooth kernel density function $g(\mathbf{p})$ is defined to reflect the probability that a point $\mathbf{p} \in \mathbb{R}^3$ is a point on the surface S sampled by a noisy point cloud Ω. Inspired by the previous work of Schall et al. [21], we measure the probability density function $g(\mathbf{p})$ by considering the squared distance of $\mathbf{p}$ to the plane $H(\mathbf{p})$ fitted to a spatial k-neighborhood of $\mathbf{p}_i$ as

$$g(\mathbf{p}) = \sum_{i=1}^{L} g_i(\mathbf{p}) = \sum_{i=1}^{L} \Phi_i(\mathbf{p} - \mathbf{p}_{\text{pro}}) G_i(\mathbf{p}_{\text{pro}} - \bar{\mathbf{p}}_i) \left\{ 1 - \left[(\mathbf{p} - \bar{\mathbf{p}}_i) \cdot \mathbf{n}_i / h \right]^2 \right\}, \quad (3)$$

where Φ_i and G_i are two monotonically decreasing weighting functions to measure the spatial distribution of point samples from spatial domain and range domain, which are more adaptive to the local geometry of the point model. The weight function could be either a Gaussian kernel or an Epanechnikov kernel. Here we choose Gaussian function $e^{-x^2/2\sigma^2}$. The $\mathbf{p}_{\text{pro}}$ is an orthogonal projection of a certain sample point $\mathbf{p}$ on the least-square fitting plane. The positions $\mathbf{p}$ close to $H(\mathbf{p})$ will be assigned with a higher probability than the positions being more distant.

The simplest method to find the local maxima of (3) is to use a gradient-ascent process written as follows:

$$\nabla g(\mathbf{p}) = \sum_{i=1}^{L} \nabla g_i(\mathbf{p}) \approx \frac{-2}{h^2} \sum_{i=1}^{L} \Phi_i(\mathbf{p} - \mathbf{p}_{\text{pro}}) G_i(\mathbf{p}_{\text{pro}} - \bar{\mathbf{p}}_i) \left[(\mathbf{p} - \bar{\mathbf{p}}_i) \cdot \mathbf{n}_i \right] \cdot \mathbf{n}_i . \quad (4)$$

Thus the mean shift vectors are determined as

$$m(\mathbf{p}) = \mathbf{p} - \left\{ \sum_{i=1}^{L} \Phi_i(\mathbf{p} - \mathbf{p}_{\text{pro}}) G_i(\mathbf{p}_{\text{pro}} - \bar{\mathbf{p}}_i) \left[(\mathbf{p} - \bar{\mathbf{p}}_i) \cdot \mathbf{n}_i \right] \cdot \mathbf{n}_i \Big/ \sum_{i=1}^{L} \Phi_i(\mathbf{p} - \mathbf{p}_{\text{pro}}) G_i(\mathbf{p}_{\text{pro}} - \bar{\mathbf{p}}_i) \right\}. \quad (5)$$

Combining equations (4) and (5) we get the resulting iterative equations of mean shift filtering

$$\mathbf{p}_i^{j+1} = m(\mathbf{p}_i^j), \ \mathbf{p}_i^o = \mathbf{p}_i , \quad (6)$$

where j is the number of iteration. In our algorithm, $g(\mathbf{p})$ satisfies the following conditions

$$g(\mathbf{p}_2) - g(\mathbf{p}_1) > \nabla g(\mathbf{p}_1)(\mathbf{p}_2 - \mathbf{p}_1) \quad \forall \mathbf{p}_1 \geq 0, \forall \mathbf{p}_2 \geq 0, \quad (7)$$

thus $g(\mathbf{p})$ is a convex function with finite stable points in the set $U = \left\{ \mathbf{p}_i \mid g(\mathbf{p}_i) \geq g(\mathbf{p}_i^1) \right\}$ resulting in the convergence of the series $\left\{ \mathbf{p}_i^j, i = 1, ..., L, j = 1, 2, ... \right\}$. Experiments show that we stop iterative process if $\left\| \mathbf{p}_i^{j+1} - \mathbf{p}_i^j \right\| \leq 5 \times 10^{-3} h$ is satisfied. Each sample usually converges in less than 8 iterations. Due to the clustering property of our method, groups of outliers usually converge to a set of single points sparsely distributed around the surface samples. These points can be characterized by a very low spatial sampling density compared to the surface samples. We use this criteria for the detection of outliers and remove them using a simple threshold.

3 Implicit Surface Reconstruction

3.1 Adaptive Space Subdivision

In order to avoid solving a dense linear system, we subdivide the whole input points filtered by mean shift into slightly overlapping subdomains. An adaptive octree-based subdivision method introduced by Ohtake et al. [18] is used in our space partition.

We define the local support radius $R=\alpha\ d_i$ for the cubic cells which are generated during the subdivision, d_i is the length of the main diagonal of the cell. Assume each cell should contain points between T_{min} and T_{max}. In our implementation, α=0.6, T_{min} =20 and T_{max} =40 has provided satisfying results.

A local max-norm approximation error is estimated according to the Taubin distance [19],

$$\varepsilon = \max_{|\mathbf{p}_i - c_i| < R} \left| f(\mathbf{p}_i) \right| / \left| \nabla f(\mathbf{p}_i) \right| . \tag{8}$$

If the ε is greater than a user-specified threshold ε_0, the cell is subdivided and a local neighborhood function f_i is built for each leaf cell.

3.2 Estimating Local Shape Functions

Given the set of N pairwise distinct points $\Omega=\{\mathbf{p}_i\}_{i\in[1,N]}$, $\mathbf{p}_i \in\mathbb{R}^3$, which is filtered by mean shift algorithm, and the set of corresponding values $\{v_i\}_{i\in[1,N]}$, $v_i \in\mathbb{R}$, we want to find an interpolation $f: \mathbb{R}^3\rightarrow\mathbb{R}$ such that

$$f(\mathbf{p}_i) = v_i . \tag{9}$$

We choose the $f(\mathbf{p})$ to be a radial basis function of the form

$$f(\mathbf{p}) = \eta(\mathbf{p}) + \sum_{i=1}^{N} \omega_i \varphi\left(|\mathbf{p} - \mathbf{p}_i|\right), \tag{10}$$

where $\eta(\mathbf{p})= \zeta_k \eta_k(\mathbf{p})$ with $\{\eta_k(\mathbf{p})\}_{k\in[1,Q]}$ is a basis in the 3D null space containing all real-value polynomials in 3 variables and of order at most m with $Q = \binom{m+3}{3}$ depending on the choice of φ, φ is a basis function, ω_i are the weights in real numbers, and $|\ .\ |$ denotes the Euclidean norm.

There are many popular basis functions φ for use: biharmonic $\varphi(r) = r$, triharmonic $\varphi(r) = r^3$, multiquadric $\varphi(r) = (r^2+c^2)^{1/2}$, Gaussian $\varphi(r) = \exp(-cr^2)$, and thin-plate spline $\varphi(r) = r^2\log(r)$, where $r = |\mathbf{p}-\mathbf{p}_i|$.

As we have an under-determined system with $N+Q$ unknowns and N equations, so-called natural additional constraints for the coefficient ω_i are added in order to ensure orthogonality, so that

$$\sum_{i=1}^{N} \omega_i \eta_1 = \sum_{i=1}^{N} \omega_i \eta_2 = \cdots = \sum_{i=1}^{N} \omega_i \eta_Q = 0 . \tag{11}$$

The equations (9), (10) and (11) may be written in matrix form as

$$\begin{pmatrix} \mathbf{A} & \boldsymbol{\eta} \\ \boldsymbol{\eta}^{\mathrm{T}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\omega} \\ \boldsymbol{\zeta} \end{pmatrix} = \begin{pmatrix} \mathbf{v} \\ 0 \end{pmatrix}, \tag{12}$$

where $\mathbf{A}=\varphi(|\mathbf{p}_i-\mathbf{p}_j|)$, i,j =1,…,N, $\boldsymbol{\eta}=\eta_k(\mathbf{p}_i)$, i=1,…,N, k=1,…,Q, $\boldsymbol{\omega}=\omega_i$, i=1,…,N and $\boldsymbol{\zeta}=\zeta_k$, k=1,…,Q. Solving the linear system (14) determines ω_i and ζ_k, hence the $f(\mathbf{p})$.

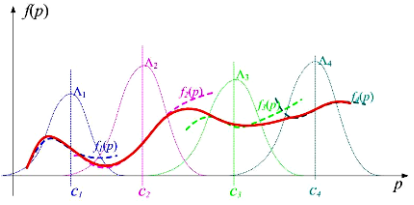


Fig. 1. A set of locally defined functions are blent by the PoU method. The resulting function (red solid curve) is constructed from four local functions (thick dashed curves) with their associated weight functions (dashed dotted curves).

3.3 Partition of Unity

After suppressing high frequency noise and removing outliers, we divide the global domain $\Omega=\{\mathbf{p}_i\}_{i\in[1,N]}$ into M lightly overlapping subdomains $\{\Omega_i\}_{i\in[1,M]}$ with $\Omega\subseteq\bigcup_i\Omega_i$ using an octree-based space partition method. On this set of subdomains $\{\Omega_i\}_{i\in[1,M]}$, we construct a partition of unity, i.e., a collection of non-negative functions $\{\Lambda_i\}_{i\in[1,M]}$ with limited support and with $\sum\Lambda_i=1$ in the entire domain Ω. For each subdomain Ω_i we construct a local reconstruction function f_i based on RBF to interpolate the sampled points. As illustrated in Fig. 1, four local functions $f_1(\mathbf{p})$, $f_2(\mathbf{p})$, $f_3(\mathbf{p})$ and $f_4(\mathbf{p})$ are blended together by weight functions Λ_1, Λ_2, Λ_3 and Λ_4. The red solid curve is the final reconstructed function.

Now an approximation of a function $f(\mathbf{p})$ defined on Ω is given by a combination of the local functions

$$f(\mathbf{p})=\sum_{i=1}^{M} f_i(\mathbf{p})\Lambda_i(\mathbf{p})\cdot \tag{13}$$

The blending function is obtained from any other set of smooth functions by a normalization procedure

$$\Lambda_i(\mathbf{p})=w_i(\mathbf{p})\Big/\sum_j w_j(\mathbf{p})\cdot \tag{14}$$

The weight functions w_i must be continuous at the boundary of the subdomains Ω_i. Tobor et al. [15] suggested that the weight functions w_i be defined as the composition of a distance function $D_i:\mathbb{R}^n\rightarrow[0,1]$, where $D_i(\mathbf{p})=1$ at the boundary of Ω_i and a decay function θ: $[0,1]\rightarrow[0,1]$, i.e. $w_i(\mathbf{p})=\theta\circ D_i(\mathbf{p})$. More details about D_i and θ can be found in Tobor's paper.

4 Applications and Results

All results presented in this paper are performed on a 2.8GHz Intel Pentium4 PC with 512M of RAM running Windows XP.

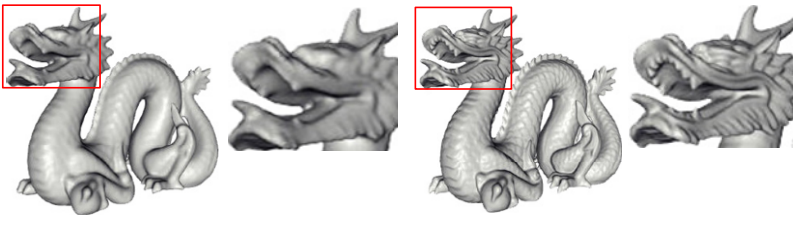
To visualize the resulting implicit surfaces, we used a pure point-based surface rendering algorithm such as [22] instead of traditionally rendering the implicit surfaces using a Marching Cubes algorithm [23], which inherently introduces heavy topological constraints.

Table 1. Computational time measurements for mean shift filtering and RBF+PoU surface reconstructing with error bounded at 10^{-5}. Timings are listed as minutes:seconds.

model	Bunny	Dragon head	Dragon
P_{input}	362K	485K	2.11M
P_{filter}	165K	182K	784K
T_{filter}	9:07	13:26	41:17
T_{octree}	0:02	0:04	0:10
T_{rec}	0:39	0:51	3:42

(a) (b) (c)

Fig. 2. Comparison of implicit surface reconstruction based on RBF methods. (a) Input noisy point set of Stanford bunny (362K). (b) Reconstruction with Carr's method [11]. (c) Reconstruction with our method in this paper.



(a) (b) (c) (d)

Fig. 3. Error threshold controls reconstruction accuracy and smoothness of the scanned dragon model consisting of 2.11M noisy points. (a) Reconstructing with error threshold at $8.4\text{x}10^{-4}$. (c) Reconstructing with error threshold at $2.1\text{x}10^{-5}$. (b) and (d) are close-ups of the red rectangle areas of (a) and (c) respectively.

Table 1 presents computational time measurements for filtering and reconstructing of three scan models, bunny, dragon head and dragon, with user-specified error threshold 10^{-5} in this paper. In order to achieve good effects of denoising we choose a large number of k-neighborhood for the adaptive kernel computation, however, more timings of filtering are spent . In this paper, we set k=200. Note that the filtered points are less than input noisy points due to the clustering property of our method.

In Fig. 2 two visual examples of the reconstruction by Carr's method [11] and our algorithm are shown. Carr et al. use polyharmonic RBFs to reconstruct smooth, manifold surfaces from point cloud data and their work is considered as an excellent and successful research in this field. However, because of sensitivity to noise, the

reconstructed model in the middle of Fig. 2 shows spurious surface sheets. The quality of the reconstruction is highly satisfactory, as be illustrated in the right of Fig. 2, since a mean shift operator is introduced to deal with noise in our algorithm.

For the purpose of illustrating the influence of error thresholds on reconstruction accuracy and smoothness, we set two different error thresholds on the reconstruction of the scanned dragon model, as demonstrated by Fig. 3.

5 Conclusions and Future Work

In this study, we have presented a robust method for implicit surface reconstruction from scattered point clouds with noise and outliers. Mean shift method filters the raw scanned data and then the PoU scheme blends the local shape functions defined by RBF to approximate the whole surface of real objects.

We are also investigating various other directions of future work. First, we are trying to improve the space partition method. We think that the Volume-Surface Tree [20], an alternative hierarchical space subdivision scheme providing efficient and accurate surface-based hierarchical clustering via a combination of a global 3D decomposition at coarse subdivision levels, and a local 2D decomposition at fine levels near the surface may be useful. Second, we are planning to combine our method with some feature extraction procedures in order to adapt it for processing very incomplete data.

Acknowledgements. This work was supported by 'Qing Lan' Talent Engineering Funds by Lanzhou Jiaotong University.

References

1. Weiss, V., Andor, L., Renner, G., Varady, T.: Advanced Surface Fitting Techniques. Computer Aided Geometric Design 1, 19–42 (2002)
2. Iglesias, A., Echevarría, G., Gálvez, A.: Functional Networks for B-spline Surface Reconstruction. Future Generation Computer Systems 8, 1337–1353 (2004)
3. Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., Silva, C.T.: Point Set Surfaces. In: Proceedings of IEEE Visualization, San Diego, CA, USA, pp. 21–28 (2001)
4. Amenta, N., Kil, Y.J.: Defining Point-Set Surfaces. ACM Transactions on Graphics 3, 264–270 (2004)
5. Levin, D.: Mesh-Independent Surface Interpolation. In: Geometric Modeling for Scientific Visualization, pp. 37–49. Springer, Heidelberg (2003)
6. Fleishman, S., Cohen-Or, D., Silva, C.T.: Robust Moving Least-Squares Fitting with Sharp Features. ACM Transactions on Graphics 3, 544–552 (2005)
7. Savchenko, V.V., Pasko, A., Okunev, O.G., Kunii, T.L.: Function Representation of Solids Reconstructed from Scattered Surface Points and Contours. Computer Graphics Forum 4, 181–188 (1995)
8. Turk, G., O'Brien, J.: Variational Implicit Surfaces. Technical Report GIT-GVU-99-15, Georgia Institute of Technology (1998)
9. Wendland, H.: Piecewise Polynomial, Positive Definite and Compactly Supported Radial Functions of Minimal Degree. Advances in Computational Mathematics, pp. 389–396 (1995)

10. Morse, B.S., Yoo, T.S., Rheingans, P., Chen, D.T., Subramanian, K.R.: Interpolating Implicit Surfaces from Scattered Surface Data Using Compactly Supported Radial Basis Functions. In: Proceedings of Shape Modeling International, Genoa, Italy, pp. 89–98 (2001)
11. Carr, J.C., Beatson, R.K., Cherrie, J.B., Mitchell, T.J., Fright, W.R., McCallum, B.C., Evans, T.R.: Reconstruction and Representation of 3D Objects with Radial Basis Functions. In: Proceedings of ACM Siggraph 2001, Los Angeles, CA, USA, pp. 67–76 (2001)
12. Beatson, R.K.: Fast Evaluation of Radial Basis Functions: Methods for Two-Dimensional Polyharmonic Splines. IMA Journal of Numerical Analysis 3, 343–372 (1997)
13. Wu, X., Wang, M.Y., Xia, Q.: Implicit Fitting and Smoothing Using Radial Basis Functions with Partition of Unity. In: Proceedings of 9th International Computer-Aided-Design and Computer Graphics Conference, Hong Kong, China, pp. 351–360 (2005)
14. Ohtake, Y., Belyaev, A., Seidel, H.P.: Multi-scale Approach to 3D Scattered Data Interpolation with Compactly Supported Basis Functions. In: Proceedings of Shape Modeling International, Seoul, Korea, pp. 153–161 (2003)
15. Tobor, I., Reuter, P., Schlick, C.: Multi-scale Reconstruction of Implicit Surfaces with Attributes from Large Unorganized Point Sets. In: Proceedings of Shape Modeling International, Genova, Italy, pp. 19–30 (2004)
16. Comaniciu, D., Meer, P.: Mean Shift: A Robust Approach toward Feature Space Analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 5, 603–619 (2002)
17. Cheng, Y.Z.: Mean Shift, Mode Seeking, and Clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence 8, 790–799 (1995)
18. Ohtake, Y., Belyaev, A., Alexa, M., Turk, G., Seidel, H.P.: Multi-level Partition of Unity Implicits. ACM Transactions on Graphics 3, 463–470 (2003)
19. Taubin, G.: Estimation of Planar Curves, Surfaces and Nonplanar Space Curves Defined by Implicit Equations, with Applications to Edge and Range Image Segmentation. IEEE Transaction on Pattern Analysis and Machine Intelligence 11, 1115–1138 (1991)
20. Boubekeur, T., Heidrich, W., Granier, X., Schlick, C.: Volume-Surface Trees. Computer Graphics Forum 3, 399–406 (2006)
21. Schall, O., Belyaev, A., Seidel, H.-P.: Robust Filtering of Noisy Scattered Point Data. In: IEEE Symposium on Point-Based Graphics, Stony Brook, New York, USA, pp. 71–77 (2005)
22. Rusinkiewicz, S., Levoy, M.: Qsplat: A Multiresolution Point Rendering System for Large Meshes. In: Proceedings of ACM Siggraph 2000, New Orleans, Louisiana, USA, pp. 343–352 (2000)
23. Lorensen, W.E., Cline, H.F.: Marching Cubes: A High Resolution 3D Surface Construction Algorithm. Computer Graphics 4, 163–169 (1987)
24. Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W.: Surface Reconstruction from Unorganized Points. In: Proceedings of ACM Siggraph 1992, Chicago, Illinois, USA, pp. 71–78 (1992)

A Discrete Approach to Compute Terrain Morphology

Paola Magillo[1], Emanuele Danovaro[2], Leila De Floriani[1],
Laura Papaleo[1], and Maria Vitali[1]

[1] Department of Information and Computer Science, University of Genova, Italy
{magillo,deflo,papaleo,vitali}@disi.unige.it
http://www.disi.unige.it
[2] Free University of Bozen–Bolzano, Italy
emanuele.danovaro@unibz.it
http://www.unibz.it

Abstract. We consider the problem of extracting morphology of a terrain represented as a Triangulated Irregular Network (TIN). We propose a new algorithm and compare it with representative algorithms of the main approaches existing in the literature to this problem. The new algorithm has the advantage of being simple, using only comparisons (and no floating-point computations), and of being suitable for an extension to higher dimensions. Our experiments consider both real data and artificial test data. We evaluate the difference in the results produced on the same terrain data, as well as the impact of resolution level on such a difference, by considering representations of the same terrain at different resolutions.

Keywords: Morse and Morse-Smale Decomposition, Terrain Morphology, Terrain Analysis.

1 Introduction

Extracting and representing morphological information is a very relevant issue in order to develop automatic tools for gaining and maintaining knowledge of terrain models which are are widely used in different application contexts such as Geographic Information Systems (GISs), Virtual Reality, Entertainment and so on.

A terrain model is a scalar field, i.e., a function $f(x,y)$ (usually called height function) defined on a domain D. Often, f is known only at a finite set of sampled points and it is approximated through a discrete digital model: a *Regular Square Grid* (RSG) if the sampled points are regularly spaced, and a *Triangulated Irregular Network* (TIN) if they are irregularly sampled. Both RSGs and TINs provide accurate representations of terrains, but they fail in capturing the morphological structure defined by critical points (pits, peaks, passes), and integral lines (ridges, valleys). On the contrary, a morphological terrain description is compact and supports a knowledge-based approach to analyze, visualize and understand a terrain dataset, as required, for instance, in visual data mining applications.

In the last decades, there has been a lot of research focusing on extracting critical features (points, lines or regions) from images or terrain data described by an RSG, or a TIN. More recent works in computational geometry concentrate on representing

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 13–26, 2008.

the morphology of terrains through a decomposition of the terrain surface into regions bounded by critical points (minima, maxima, saddle points) and integral lines. These techniques are rooted in Morse theory and try to simulate the decomposition of a terrain induced by C^2 Morse functions in the discrete case.

In this paper, we propose a new algorithm for extracting morphological information (in the form of the stable and unstable Morse complexes) from a terrain model described by a TIN, which is simple, requires no floating point calculations, and can manage special configurations such as flat triangles and edges. We also present a comprehensive study of analogous existing methods and propose a set of experiments in order to evaluate our approach.

Recall that a TIN basically consists of a triangulation Σ covering the field domain D of the height function f, having its vertices at the sampled points. In a *triangulation*, two nearby triangles can only touch each other by sharing a vertex, or a common edge. On each triangle t in Σ, function f is approximated as a linear interpolant of the height values sampled at the three vertices of t. Note that RSGs can be reduced to TINs by triangulating each square into two triangles.

In the remainder of this paper, Section 2 introduces some basic background notions; Section 3 discusses related works; Section 4 presents our novel algorithm; Section 5 introduces three representative algorithms that we have implemented for comparison, and Section 6 presents an experimental evaluation of our novel algorithm compared to these three methods. Finally, Section 7 draws some concluding remarks.

2 Background

Morse theory is a powerful tool to capture the topological structure of a scalar field in the continuum [15]. Let f be a C^2 real-valued function defined over a domain $D \subseteq \mathbb{R}^2$. A point $p \in \mathbb{R}^2$ is called a *critical point* of f if and only if the *gradient* of f vanishes at p. The function f is a *Morse function* if and only if the Hessian matrix $H_p f$ of the second derivatives of f at a critical point p is non-singular (its determinant is $\neq 0$): basically, if all its critical points are non-degenerate. This implies that the critical points of a Morse function are *isolated*. The number of negative eigenvalues of $H_p f$ is called the *index* of a critical point p. In 2D, a non-degenerate critical point p of a Morse function f can be of three types: a *minimum* (*pit*), a *saddle*, or a *maximum* (*peak*), if p has index 0, 1 or 2, respectively. An *integral line* of a function f is a maximal path which is everywhere tangent to the gradient vector field (see Figure 1 (a)). It is emanating from a critical point or from the boundary of D, and it reaches another critical point or the boundary of D. An integral line which connects a maximum to a saddle, or a minimum to a saddle, is called a *separatrix line*. In Geographic Information Systems (GISs), separatrix lines that connect minima to saddles are usually called *ravine*, or *valley lines*, while those that connect saddles to maxima are called *ridge lines*.

Integral lines that converge to a maximum, a saddle and a minimum form a 2-dimensional (region), 1-dimensional (line) and 0-dimensional (point) cell, respectively, and they are called *unstable manifolds*. Integral lines that originate from a minimum, a saddle and a maximum form a 2-, 1- and 0-dimensional cell, respectively, and they are called *stable manifolds*. See Figure 1 (b). The stable (unstable) manifolds are pair-wise

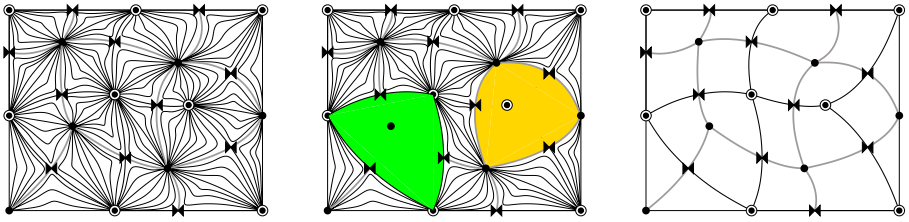


Fig. 1. (a) Integral lines, symbols •, ⊙, ⋈ denote minima, maxima and saddles, respectively. (b) The 2-manifolds corresponding to a minimum (green) and to a maximum (yellow). (c) The Morse-Smale complex, its 1-skeleton is the critical net.

disjoint cells and form a complex, since the boundary of every cell is the union of lower-dimensional cells. They are called *stable* and *unstable Morse complexes*, respectively.

A Morse function f is a *Morse-Smale function* when the stable and the unstable manifolds intersect only transversally. In two dimensions, this means that the stable and unstable 1-manifolds (lines) cross when they intersect, and the crossing points are saddles.

A *Morse-Smale complex* is the complex defined by the intersection of the stable and unstable Morse complexes for a function f which is a *Morse-Smale function*. The 1-skeleton of a Morse-Smale complex consists of the critical points and the separatrix lines joining them, and it is called a *critical net* (see Figure 1 (c)).

The *surface network* [11,14] used in Geographic Information Systems (GISs) for morphological terrain modeling, is essentially the critical net.

3 Related Work

Several algorithms have been proposed in the literature for decomposing the domain of a scalar field f (as a terrain model) into an approximation of a Morse complex (or of a Morse-Smale complex). They either fit a C^1 or C^2 surface on a terrain model, or simulate a Morse-Smale complex (a Morse complex) in the discrete case. By assuming that no two adjacent vertices in the TIN have the same height, they ensure that the critical points are isolated, as in the Case of C^2 Morse functions [7].

A Morse (Morse-Smale) complex can also be defined using the concepts related to watershed tranform [9,18,12,8,16]. The watershed transform in the C^2 case provides a decomposition of a the domain of a function f into open regions of influence associated

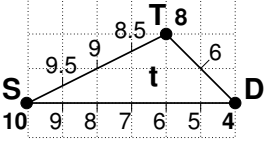


Fig. 2. Edge labelled T-D is steeper than edge labelled S-D. Numbers denote vertex heights.

to the minima, called *catchment basins*. Catchment basins can be described in terms of topographic distance [9]. In the 2D case, if the function f is a Morse function, the catchment basins of the minima are essentially 2-manifolds of the stable Morse complex. Through a change in the sign of the Morse function f, the 2-manifolds (associated to the maxima) of the unstable Morse complex can be extracted.

In order to build a structural representation of a given scalar field f, all the existing methods extract critical points of f as a first step of the global procedure. The most common approach to compute critical points examines, for each vertex p in the TIN, the neighbor points (sharing with p and edge) and computes the height difference between every point and p. If all differences are positive (p is lower than its neighbors), then p is a *minumum*. If all differences are negative (p is higher than its neighbors), then p is a *maximum*. If the number of sign changes of such difference, while traversing p's neighbors in cyclic order, is two, then p is a regular, i.e., non-critical point. If the number of sign changes is four, then p is a *saddle*; if it is more than four, then p is a multiple saddle. This technique is used by almost all the algorithms, with the exception of [2].

Existing algorithms for extracting an approximation of a Morse (Morse-Smale) complex can be classified according to: the input they consider (namely RSG or TIN), the output they produce (namely an approximation of a Morse-Smale complex or of a Morse complex) and the algorithmic technique they choose. Here, we have classified them into *boundary-based* or *region-based* techniques [4].

Boundary-based techniques basically extract an approximation of the critical net, by computing the critical points and then tracing the integral lines, starting from saddle points [1,13,17,7,2,3,10]. *Region-based techniques* extract a discrete approximation of the stable and unstable Morse complexes, by starting from minima and maxima and letting a region grow until a given condition is reached [5,6,9,18,8]. We included watershed algorithms in the latter class since they are region-based in nature.

This paper is organized as follows. In Section 5 we present our implementations of some representative algorithms of the above techniques. All algorithms, with the exception of the watershed approach, require that the three vertices of a triangle have distinct heights. This is generally achieved, when necessary, by perturbation of the height values.

4 The STD Algorithm

In this section we present our novel algorithm, that we called *STD*, for extracting the 2-manifolds (i.e., cells associated with the minima) of a stable Morse complex for a (Morse) function f defined on a TIN. The algorithm is region-based in nature since it starts from the minima and lets the 2-manifolds of the Morse complex grow as long as it is possible.

We first describe the algorithm under the assumption that no two vertices of the terrain have the same height. Successively, we relax this assumption and show how to deal with flat triangles, and triangles having one flat edge.

4.1 Basic Version of the Algorithm

The STD algorithm performs three main steps:

1. **Classify** the vertices of each triangle t in the TIN, based on their heights.
2. **Extract** the minima of the function in the TIN.
3. **For each** minimum p, **construct** the stable 2-manifold by iteratively adding triangles to it.

Vertex classification and Extraction of Local Minima. For each triangle t in the TIN, the highest, middle, and lowest vertex are labeled as *Source* (S), *Through* (T), and *Drain* (D), respectively.

By this STD configuration of the vertices we basically simulate the gradient direction of t in the discrete case. Note that this labelling does not assume any kind of interpolation (linear or higher-order) on triangles or edges of the mesh. Edge labelled S-D is not necessarily the edge of steepest descent. In Figure 2 the steepest descent is at edge labelled T-D.

The local minima identification is very simple: they are found as those vertices labeled D in all their incident triangles.

Construction of the Stable 2-Manifolds. For each minimum p, the stable 2-manifold γ_p associated with p is initialized with all triangles of the TIN which are incident in p. Successively, an iterative phase starts in which, at each step, the algorithm decides if a triangle t, externally adjacent to one edge e of the current perimeter of γ_p, can be added to γ_p. The rationale for this decision takes the following issues into account: *(i)* the choice must reflect the intuition that water flows from a higher to a lower height, *(ii)* the choice must be deterministic, i.e., a triangle t cannot be included into different 2-manifolds, depending on the order in which minima are processed.

The algorithm maintains the invariant that, if a triangle t has been included into γ_p, then the edge of t labelled T-D is not on the boundary of γ_p.

4.2 Inclusion of a Triangle

Let e be an edge of the current perimeter of γ_p, and t be the triangle externally adjacent to e. The decision whether to include t into γ_p or not, is based on the STD configuration of its vertices. There are three possible cases.

Case 1. If the vertex v of t opposite to e is labelled D in t, then we do not include t into γ_p. See Figure 3 (a). This is according to the intuition that water cannot exit t through e, since it naturally flows towards v. Triangle t will be included when we will reach it from another edge, and Case 2 or 3 will hold.

Case 2. If the vertex v of t opposite to e is labelled S in t, then we include t into γ_p. See Figure 3 (b). Intuitively, water tends to flow across t and reach vertex v', endpoint of e, which is labelled D in t. The question is whether it will exit t through e (in that case t belongs to γ_p) or through the edge of t labelled S-D. Now, we explain why we have decided that water passes through edge e.

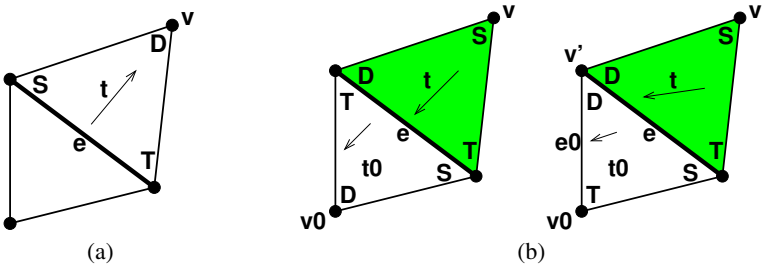


Fig. 3. Case 1 (a) and Case 2 (b). Arrows denote water flow. Green triangles are included.

Let t_0 be the triangle belonging to γ_p and adjacent to t along e, and let v_0 be the vertex of t_0 opposite to e. Note that, for the invariant, e cannot be labelled T-D in t_0 (equivalently, v_0 cannot be labelled S).

If e is labelled S-T in t_0, then water enters t_0 through e, therefore it must exit from t through e.

If e is labelled S-D in t_0, then water exits t_0 through its edge e_0 labelled T-D (it cannot exit through the other edge, since it is labelled S-T, and it must exit from one edge different from e otherwise t_0 would not have been included in γ_p). Therefore water that flows across t and reaches vertex v (which is labelled D in both t and t_0) turns around v', enters t_0, and finally exits t_0 through e_0.

Note that the invariant is maintained: edge e (labelled T-D in the newly included triangle t) is inside the updated 2-manifold γ_p.

Case 3. If the vertex v of t opposite to e is labelled T in t, then the situation is more complex. Certainly, water flows to vertex v', endpoint of e, which is labelled D in t. Then, will it exit from t into γ_p through edge e, or will it exit t through its edge e' labelled T-D, towards the 2-manifold existing on the other side?

Starting from t, we explore the maximal fan of triangles having their lowest vertex in v' (i.e., v' is labelled D in all such triangles). Let w be the vertex of maximum height among the vertices of such triangles. The part of the fan starting from t and going up to edge $v'w$ is included into γ_p. See Figure 4 (a). The other part of the fan will be later included into the 2-manifold existing on the other side. Note that, if w is the same as the vertex labelled S in t, then no triangle is included. See Figure 4 (b).

The invariant is maintained since the edges remaining on the boundary of the updated 2-manifold γ_p are $v'w$, and edges opposite to v': none of them is labelled T-D. In fact, edges opposite to v' are labelled S-T in the just included triangles, and edge $v'w$ is labelled S-D in both adjacent triangles.

Note that the management of Case 3 does not interfere with Case 2. In fact, the edge e_1 marking the other side of the fan may be labelled T-D in its adjacent triangle t_1 belonging to the fan. In this case, when reached from e_1, t_1 will be included into the 2-manifold γ_q existing on the other side of e_1. The triangle adjacent to t_1 along the other edge of t_1 incident in v' may be in the same situation (and thus be included in γ_q as well), and so on. Thus, a whole fan of triangles, starting from t_1, is included into γ_q. But this fan must end at edge w, because the opposite vertex to $v'w$ is labelled T in the next

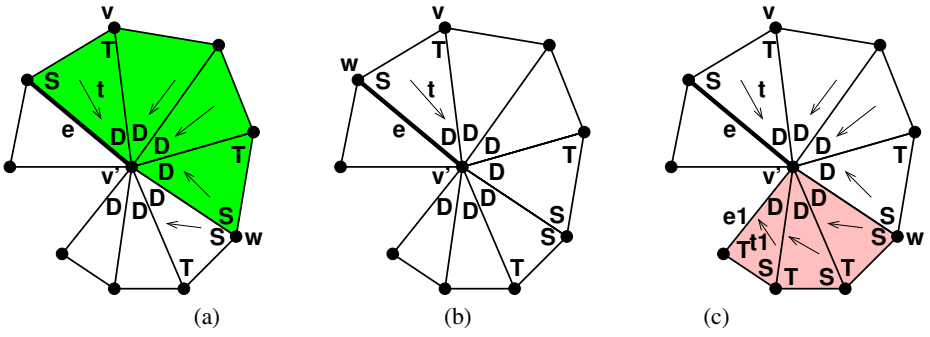


Fig. 4. (a) Case 3 with non-empty set of included triangles; green triangles are included. (b) Case 3 with empty the set of included triangles. (c) Inclusion of the remaining triangles of the fan by applying Case 2 from edge e_1.

triangle. Thus, there is no interference between Case 3 applied from edge e, and Case 2 repeatedly applied starting from edge e_1. See Figure 4 (c).

4.3 Time Complexity

It can be easily shown that every triangle t of the TIN is examined at most three times, one from each edge, before being included into some 2-manifold. Thus, the worst-case time complexity of our algorithm is $O(n)$ where n is the number of TIN vertices. The only non-trivial part in this statement is showing that, in Case 3, a triangle can be in a traversed fan, without being included, at most once during the whole algorithm. The triangles of the fan, which are not included, are those located beyond edge $v'w$. The same fan may be traversed from the opposite side, while growing another 2-manifold γ_q. Since we will be traversing the same fan in the opposite way, in that situation exactly those triangles, that were not previously included, will be found before edge $v'w$, and will be included into γ_q.

4.4 Management of Special Cases

Now, we explain how the STD algorithm deals with flat triangles, and triangles with a flat edge.

In a preprocessing step, we find edge-connected areas of flat triangles, and vertex-connected networks of flat edges that are not edge- or vertex-incident into a flat triangle. Such areas / networks are candidate to act as 1- or 2-dimensional local minima. Let h be the height of a flat area or network. Let h' be the minimum height of the third vertices of triangles externally adjacent to the perimeter of the flat area, or incident into edges of the network. If $h' > h$ then the flat area / network is treated as a local minimum (see Figure 5 (a) and (b)): its 2-manifold is initialized with all the triangles of the flat area, or with all triangles incident in the flat network, and it is expanded in the same way as other 2-manifolds.

A flat area that is not a local minimum (i.e., $h' < h$) is assigned to the 2-manifold containing the triangle t', externally adjacent to the flat area, whose third vertex has height

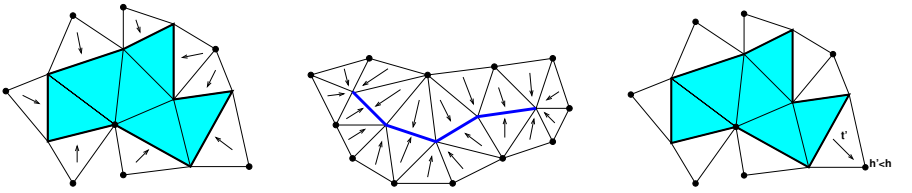


Fig. 5. Connected sets of flat triangles and edges (colored), with their adjacent triangles. Flat area (a) and flat network (b) act as local minima, unlike (c). Arrows denote water flow.

h' (see Figure 5 (c)). If t' is not unique, then we choose the 2-manifold corresponding to the lowest local minimum (if unique), or arbitrarily (otherwise).

During the algorithm, triangles with a flat edge may be examined to test whether they can be included into a growing 2-manifold. For such purpose, Cases 1, 2, and 3 introduce some exceptions when triangle t has a flat edge.

An exception may arise in Case 1, when the opposite vertex v, labelled D, is endpoint of the flat edge of t. In this case, we consider triangle t' which is adjacent to t along its flat edge e'. See Figure 6 (a). If edge e' is higher than the third vertex of t', we do not include t (no exception).

If edge e' is lower than the third vertex of t', then this is an exception: we construct the fan of triangles incident into the vertex of t which is labelled D, and proceed in the same way as in Case 3. In fact, triangles t and t' can only be included together from one of their edges labelled S-D.

Another exception arises in Case 2, when the opposite vertex v, labelled S, is endpoint of the flat edge of t. In this case, the two non-flat edges of t, e and e', are in the same situation. We must decide whether to include t into γ_p from e, or to include t into the 2-manifold that will reach t from edge e'. We construct the fan of triangles incident into the vertex of t which is labelled D, and proceed as in Case 3. See Figure 6 (b).

In Case 3, the constructed fan cannot include flat triangles, and cannot include triangles with a flat edge, when the flat edge belongs to a local minimum network. If we find one of these cases, then we stop extending the fan. See Figure 6 (c).

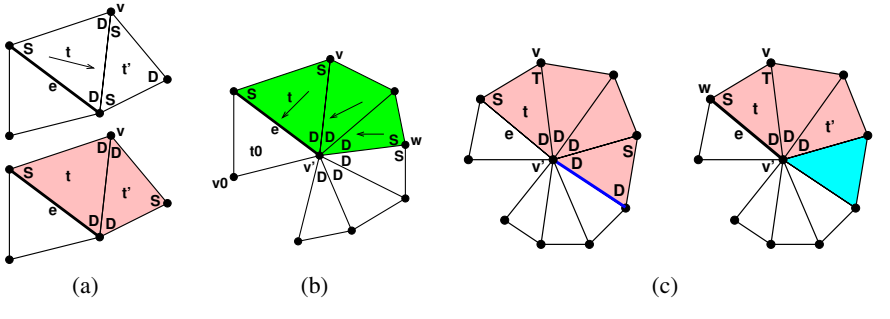


Fig. 6. Processing triangles with flat edges. Arrows denote water flow. (a) Triangle t is not included from edge e, pink triangles t and t' are processed as in Case 3. (b) Green triangles are included from edge e. (c) Construction of the fan encounters a flat edge (blue) which belong to a local minimum net, and a flat triangle (cyan): the fan will include only the pink triangles.

Again in Case 3, the procedure described in Section 4.2 takes the edge $v'w$, connecting the center v' of the fan with its upper point w, as the edge where to split the fan and assign its triangles to the 2-manifolds existing on the two sides of the fan (see Figure 4 (c)). Now, vertex w of maximum height may not be unique. Let w_1, w_2, ... w_M ($M > 1$) be the vertices having the maximum height, sorted in counterclockwise order along the fan. We split the fan at edge $v'w_i$ where i is the integer result of division $M/2$.

5 Representative Morphology Algorithms

We have implemented a number of algorithms that we have chosen as representative of the approaches existing in the literature (see Section 3).

5.1 A Boundary-Based Algorithm

Our implementation of a boundary-based approach is inspired by [7,17]. It extracts the Morse-Smale complex from a TIN by computing the critical net, in two basic steps:

1. **Extract** the critical points and **unfold** multiple saddles.
2. **Compute the 1-cells** of the complex by starting from the saddle points, and tracing two paths of steepest descent and two paths of steepest ascent, which stop at minima and maxima, respectively.

Starting from each (simple) saddle p, the algorithm computes the four lines belonging to the critical net which are incident in p. At each step, the path is extended by adding the edge corresponding to the maximum positive [negative] slope, until a maximum [minimum] is found. In the implementation we present in this paper we refer only to the stable Morse complex: for each saddle we trace *two* lines which follow the maximum positive slope and stop when two maxima are found.

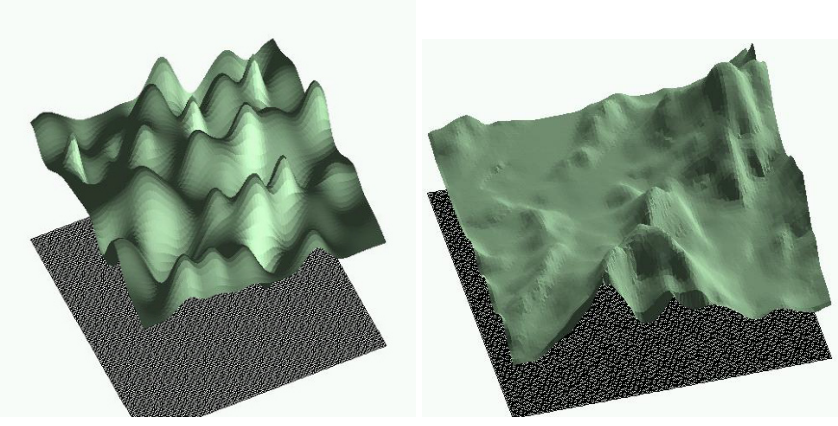
5.2 A Region-Based Algorithm

We have presented in [5] an algorithm for computing both the stable and unstable Morse complexes for a TIN. The algorithm can be sketched into two main steps:

1. **Extract** minima and maxima.
2. **Compute** the stable (unstable) Morse complex by applying a region-growing procedure. This procedure adds triangles to a 2-manifold iteratively.

For extracting the stable Morse complex, the algorithm computes the *gradient* for each triangle t in M, and the angles between the gradient and the normal vector at each edge of t (pointing outwards from the triangle). The edges of t corresponding to the largest and to the smallest angle are marked as *exit* and *entrance*, respectively.

A 2-manifold γ_p of the stable complex is initialized with the triangles incident in a local minimum p. At a generic step, γ_p is extended by adding a new triangle t sharing an edge e with γ_p, provided that e is an entrance for t and an exit for the triangle t' in γ_p sharing edge e with t. The unstable complex is computed in a completely symmetric way.



EGGS (6561 vertices) MARCY (1000 vertices)

Fig. 7. Two of the test TINs

5.3 A Watershed Algorithm

We have implemented the watershed algorithm based on simulated immersion [18]. Our implementation is applicable to TINs with flat edges and/or flat triangles and it consists of mainly three macro-steps:

1. **Sort** the vertices in increasing order with respect to the height value.
2. Perform the **flooding step** level by level, starting from the minima: this labels every vertex as belonging to a 2-manifold associated to a minimum.
3. **Assign** triangles to basins based on the labels of their vertices.

The flooding step assigns a distinct label to each minimum m and to the vertices of its associated 2-manifold γ_m. Those vertices, where two 2-manifolds meet are instead labeled as *watershed* vertices. At each iteration, a height value h (initially, the minimum height) is considered. All vertices with the same height h are first given a neutral label. Then those vertices whose neighbors have been labeled during the previous iteration are processed in order to assign the label of a 2-manifold γ_m to them.

To assign the label to a vertex p, we examine the neighbor vertices of p. If they all belong to the same 2-manifold γ_m, or some of them belong to γ_m and others are watershed points, then p is marked as belonging to γ_m. If they belong to two or more different 2-manifolds, then p is marked as a watershed point. The same operations is recursively repeated on the neighbor points of the just labeled vertices which have a neutral label (i.e., height $= h$).

Vertices at height h that are not connected to any previously processed vertex still have the neutral label. Such vertices belong to a set of new minima at level h, and get a new label.

Finally, we label each triangle t. If all the vertices of t, that are not watershed points, belong to the same 2-manifold γ_m, then we assign the triangle to γ_m. If two vertices belong to different 2-manifolds, then t is assigned to the 2-manifold related to the vertex with the lowest height.

6 Experiments

The goal of this section is to measure the quality of the results of the STD algorithm proposed in this paper, as well as evaluating the degree of uncertainty in morphology computation, i.e., to which extent the current algorithms are able to provide consistent results. We perform different experimental comparisons on both real and synthetic datasets by using our STD algorithm, the boundary-based (BND), the region-based (REG), and the watershed (WTS) algorithm described in Section 5.

Algorithm STD is of course very different from BND; STD and WTS have in common the idea of growing 2-manifolds from local minima; REG is similar in approach, but (i) it uses the gradient, and (ii) it builds a 2-manifold in pieces which are then glued together, while STD builds every 2-manifold directly, thanks to the mechanism of fans (Case 3).

We show results using two different terrains (see Figure 7):

- **EGGS**, a synthetic terrain built by sampling a function which is a combination of two planes and 64 gaussian surfaces,
- **MARCY** representing part of a real terrain model provided with the US Geological Survey in which heights have been perturbed in order to remove flat edges.

We have three TINs for EGGS, corresponding to different sampling resolutions (6,561, 25,921, and 103,041 vertices), and three TINs for MARCY, corresponding to approximations of the terrain at different resolutions (1,000, 5,000 and 10,000 vertices). Some images of the computed stable Morse complexes are in Figures 8 and 9, and Figure 10.

Table 1 evaluates the difference in the results between our new STD algorithm and the other three. This also provides a measure of the uncertainty of results. In general, the STD algorithm tends to be closer to the watershed method.

Table 2 reports the quantity of TIN surface whose classification results uncertain (i.e., assigned to the 2-manifold of different minima in different algorithms). The various algorithms may disagree in their results up to an extent between 0.5% and 10.5% of the total TIN surface.

It may be surprising that algorithms differ so much in their results: up to 9% of the terrain area may be assigned to four different minima by the four considered approaches. It is also difficult to judge which one is more correct, because a ground thruth

Table 1. Triangles (t) and percentage of terrain area (a) assigned to a different 2-manifold in the new STD algorithm and in one of the other three methods

# triang.		BND	REG	WTS
EGGS				
12,800	t	398	669	71
	a	3.11	5.23	0.55
51,200	t	1934	2,721	62
	a	3.78	5.31	0.12
204,800	t	14,828	14,488	112
	a	7.24	7.07	0.55

# triang.		BND	REG	WTS
MARCY				
1,910	t	107	98	39
	a	3.89	2.95	1.64
9,788	t	554	690	151
	a	4.73	6.10	1.31
19,602	t	1,802	2,066	356
	a	9.20	10.54	1.82

Table 2. Triangles (t) and percentage of terrain area (a) assigned to a unique 2-manifold, or to 2, 3 and 4 different 2-manifolds, by the four algorithms

# triang.		# of different 2-manifolds 1	2	3	4
EGGS					
12,800	t	11,963	42	397	398
	a	93.46	0.33	3.10	3.11
51,200	t	48,221	42	1,003	1,934
	a	94.18	0.08	1.96	3.78
204,800	t	184,608	48	5,316	14,828
	a	90.14	0.02	2.60	7.24

# triang.		# of different 2-manifolds 1	2	3	4
MARCY					
1,910	t	1,744	13	46	107
	a	94.18	0.64	1.31	3.87
9,788	t	8,835	56	343	554
	a	90.26	0.57	3.50	5.66
19,602	t	17,114	149	537	1,802
	a	87.31	0.76	2.74	9.19

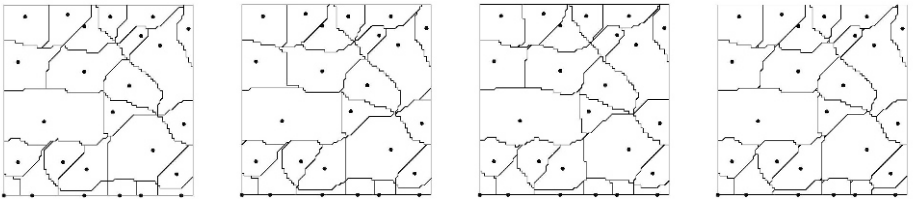


Fig. 8. The boundary of the stable Morse complex computed by the four algorithms on the EGGS terrain (6561 vertices)

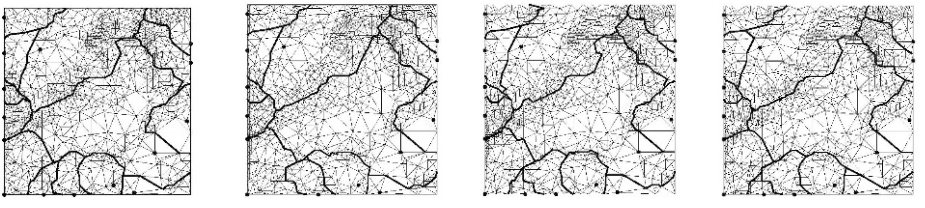
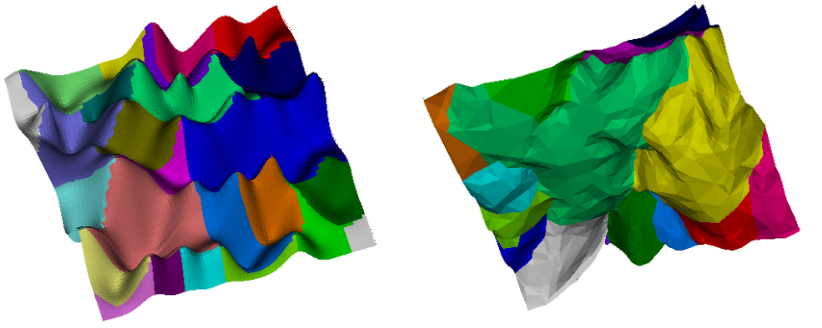


Fig. 9. The boundary of the stable Morse complex computed by the four algorithms on the MARCY terrain (1000 vertices)

is only available for C^2 functions, and not for TINs. Indeed, all existing methods only approximate Morse (or Morse-Smale) theory in the discrete case, through simplifications, conventions, and heuristics.

7 Concluding Remarks

We have proposed a new algorithm for computing the stable (unstable) Morse complex for a TIN terrain model. We performed experiments on both real and synthetic datasets in order to demonstrate the behavior of the STD algorithm with respect to other algorithms, as well as the intrinsic uncertainty of stable manifolds computation at this stage of research. We showed that our STD algorithm behaves quite well for all the



EGGS (6561 vertices) MARCY (1000 vertices)

Fig. 10. The stable Morse complex computed by STD algorith in the test TINs

test datasets and that it provides intuitively good results. Moreover, our algorithm is very simple, and requires no floating-point calculations since it uses only numerical comparisons.

Morphology algorithms that can be extended to higher dimensions have a special interest from the scientific community. Our STD algorithm is as simple as the boundary-based approach and, unlike it, seem to be more easily extensible to higher dimensions. For instance, in 3D we label the four vertices of each tetrahedron and have four cases to be managed.

Finally, [5] present a morphology-based multi-resolution terrain model, to encode different levels of approximation of a Morse-Smale complex. We plan to use the STD algorithm in this context.

Acknowledgements. This work has been partially supported by the European Network of Excellence AIM@SHAPE (contract n. 506766), by the National Science Foundation (grant CCF-0541032), by the MIUR-FIRB project SHALOM (contract n. RBIN04HWR8) and by the MIUR-PRIN project on *Multiresolution modeling of scalar fields and digital shapes*.

References

1. Bajaj, C.L., Pascucci, V., Shikore, D.R.: Visualization of scalar topology for structural enhancement. In: Proceedings IEEE Visualization, pp. 51–58. IEEE Computer Society, Los Alamitos (1998)
2. Bajaj, C.L., Shikore, D.R.: Topology preserving data simplification with error bounds. Computers and Graphics 22(1), 3–12 (1998)
3. Bremer, P.-T., Edelsbrunner, H., Hamann, B., Pascucci, V.: A multi-resolution data structure for two-dimensional Morse functions. In: Turk, G., van Wijk, J., Moorhead, R. (eds.) Proceedings IEEE Visualization, pp. 139–146. IEEE Computer Society, Los Alamitos (2003)
4. Comic, L., De Floriani, L., Papaleo, L.: Morse-Smale decomposition for modeling terrain knowledge. In: Cohn, A., Mark, D. (eds.) COSIT 2005. LNCS, vol. 3693, pp. 426–444. Springer, Heidelberg (2005)

5. Danovaro, E., De Floriani, L., Magillo, P., Mesmoudi, M.M., Puppo, E.: Morphology-driven simplification and multi-resolution modeling of terrains. In: Hoel, E., Rigaux, P. (eds.) Proceedings ACM-GIS - International Symposium on Advances in Geographic Information Systems, pp. 63–70. ACM Press, New York (2003)
6. Danovaro, E., De Floriani, L., Mesmoudi, M.M.: Topological analysis and characterization of discrete scalar fields. In: Asano, T., Klette, R., Ronse, C. (eds.) Geometry, Morphology, and Computational Imaging. LNCS, vol. 2616, pp. 386–402. Springer, Heidelberg (2003b)
7. Edelsbrunner, H., Harer, J., Zomorodian, A.: Hierarchical Morse complexes for piecewise linear 2-manifolds. In: Proceedings ACM Symposium on Computational Geometry, pp. 70–79. ACM Press, New York (2001)
8. Mangan, A., Whitaker, R.: Partitioning 3D surface meshes using watershed segmentation. IEEE Transaction on Visualization and Computer Graphics 5(4), 308–321 (1999)
9. Meyer, F.: Topographic distance and watershed lines. Signal Processing 38, 113–125 (1994)
10. Pascucci, V.: Topology diagrams of scalar fields in scientific visualization. In: Rana, S. (ed.) Topological Data Structures for Surfaces, pp. 121–129. John Wiley and Sons Ltd., Chichester (2004)
11. Pfaltz, J.L.: Surface networks. Geographical Analysis 8, 77–93 (1976)
12. Roerdink, J., Meijster, A.: The watershed transform: definitions, algorithms, and parallelization strategies. Fundamenta Informaticae 41, 187–228 (2000)
13. Schneider, B.: Extraction of hierarchical surface networks from bilinear surface patches. Geographical Analysis 37, 244–263 (2005)
14. Schneider, B., Wood, J.: Construction of metric surface networks from raster-based DEMs. In: Rana, S. (ed.) Topological Data Structures for Surfaces. John Wiley and Sons Ltd., Chichester (2004)
15. Smale, S.: Morse inequalities for a dynamical system. Bulletin of American Mathematical Society 66, 43–49 (1960)
16. Stoev, S.L., Strasser, W.: Extracting regions of interest applying a local watershed transformation. In: Proceedings IEEE Visualization, pp. 21–28. IEEE Computer Society, Los Alamitos (2000)
17. Takahashi, S., Ikeda, T., Kunii, T.L., Ueda, M.: Algorithms for extracting correct critical points and constructing topological graphs from discrete geographic elevation data. Computer Graphics Forum 14(3), 181–192 (1995)
18. Vincent, L., Soille, P.: Watershed in digital spaces: an efficient algorithm based on immersion simulation. IEEE Transactions on Pattern Analysis and Machine Intelligence 13(6), 583–598 (1991)

Procedural Natural Phenomena from Least-Cost Paths in a Weighted Graph

Ling Xu and David Mould

University of Saskatchewan, Saskatoon, Canada
{lix272,mould}@cs.usask.ca

Abstract. We present a method for creating geometric models of dendritic forms. Dendritic shapes are commonplace in the natural world; some examples of objects exhibiting dendritic shape include lichens, coral, trees, lightning, rivers, crystals, and venation patterns. Our method first generates a regular lattice with randomly weighted edges, then finds least-cost paths through the lattice. Multiple paths from a single starting location (or generator) are connected into a single dendritic shape. Alternatively, path costs can be used to segment volumes into irregular shapes. The pathfinding process is inexpensive, and allows user control through specification of endpoint placement, distribution of generators, and arrangement of nodes in the graph.

Keywords: Procedural modeling, natural phenomena, dendrites, path planning.

1 Introduction

Numerous natural phenomena, including trees, plants, lichens, coral, lightning, and river systems, can be viewed as "dendrites", a term we use in a general sense to mean any sort of branching structure. The key elements of dendritic forms are the branching structures and the erratic winding travels of individual branches. Both these characteristics can be obtained with least-cost paths in randomly weighted graphs, branching because paths to different destinations will share the early part of their route, and winding because the optimal path will have to maneuver around random expensive obstacles. In this paper, we propose explicit path planning as an algorithm for procedural creation of natural phenomena.

Proceduralism [5] is a modeling philosophy wherein models are built automatically or semi-automatically by an algorithm, with no or minimal user intervention. Procedural techniques typically revolve around the controlled use of randomness to obtain a variety of different models, all with the same kind of underlying structure. For example, fractal and multifractal terrain synthesis techniques use a random signal, and judiciously scale and sum the signal to give the illusion of a mountainside [5].

Path planning is the problem of finding the least-cost path between two nodes in a weighted graph. Algorithms for finding the least-cost path [14] are well known, since the problem appears so often in different contexts in computer science. Here, we employ path planning as an algorithm for extracting structure from randomness. We create a regular lattice with random weights on the edges; the least-cost paths through the lattice have visible structure, but since the particular weights are chosen randomly, a

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 27–40, 2008.

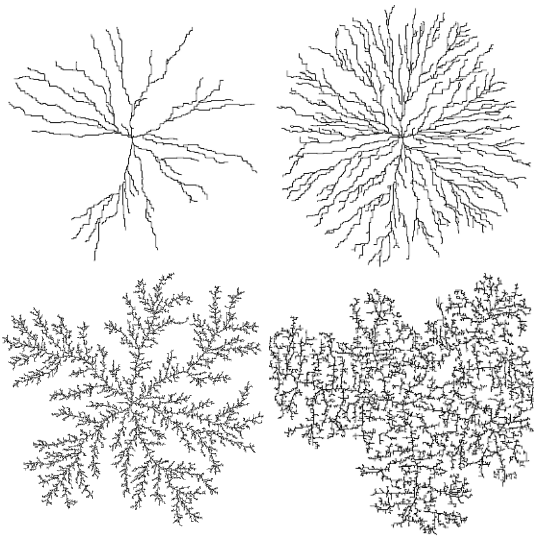


Fig. 1. Above: simple dendrites, with few or many paths; below: fractal dendrites, where new paths are repeatedly placed in the vicinity of previously chosen paths

collection of different structures can be made. Creating dendrites using path planning is straightforward: finding multiple paths from the same starting point to multiple endpoints produces a dendrite. By using an entire dendrite as the destination for a new set of paths, we can create explicitly fractal dendrites.

One of the most popular algorithmic methods for creating dendrites is diffusion-limited aggregation (DLA), first proposed in the physics literature by Witten and Sander [15]. DLA has been exploited for dendrite creation in computer graphics. However, methods for simulating DLA are slow. Employing an iterative path planning technique, we are able to produce dendrites comparable in appearance to dendritic shapes produced by a costly DLA simulation, but orders of magnitude more quickly. Some 2D dendrites created by our method appear in Figure 1.

Our basic method operates over a fixed lattice, and lattice artifacts can sometimes be seen in the resulting models, just as in lattice DLA. We also present a formulation for refining the lattice and computing a higher-resolution version of the dendrites.

We demonstrate the utility of our framework by applying it to synthesizing coral. In particular, we model staghorn coral, one of the most obviously dendritic types of coral. Our method is suitable for other types of sessile marine life, as well as for other natural objects including lichens, trees, lightning, and even terrains.

The paper is organized as follows. Following the introduction, we review some previously proposed methods for generating dendritic shapes, concentrating on DLA and L-systems. We give details of our path-planning algorithm in section 3. Results, in the form of images of dendrites and dendrite-based geometry, are shown in section 4; this section also contains timing figures. Finally, section 5 concludes our paper with a summary of the contributions and pointers to possible future work.

2 Previous Work

Algorithms for procedural geometry have been devised by computer graphics practitioners. Two algorithms in particular, L-systems and diffusion-limited aggregation, have seen considerable attention because of their versatility and the quality of their results. L-systems [10] uses a replacement grammar to create strings which can be interpreted as a variety of botanical forms, particularly (though not exclusively) branching structures. Diffusion-limited aggregation (DLA) is an algorithmic process capable of generating dendritic forms akin to those seen in a number of natural objects, including lichens, crystals, neurons, and lightning [3]. We are particularly interested in DLA, owing both to the rich set of phenomena which can be described by this method, and to the irregularities in the branching structures; we aspire to create dendrites with the same natural appearance. Other models for dendritic growth, including viscous fingering [1], the Eden model [3], and ad-hoc greedy models [6], have appeared in the physics and biology literature.

Diffusion-limited aggregation has recently been used in graphics to model lichens [4] and ice crystals [8,7]. Lightning [9] has been modeled using the related dielectric breakdown model, which describes another form of Laplacian growth. These results are of high visual quality, although the modeling process is time-consuming.

The brute-force algorithm for diffusion-limited aggregation [15] is as follows. Some initial sites in a lattice are set to "occupied"; the remainder of the lattice nodes are unoccupied. A particle is released, at a great distance from any occupied site, and undertakes a random walk until it reaches some location adjacent to an occupied site. At that point, the node where the particle is located becomes occupied, and a new walker is released. The above process is repeated, hundreds, thousands, or even millions of times. When a sufficient number of particles have been placed, the resulting aggregation has a fractal dendritic shape; the dendrites arise owing to the greater likelihood of a particle encountering the tip of a branch than a point along a branch.

L-systems is a parallel rewriting grammar that takes an initial string ("axiom") and repeatedly performs applicable transformations on it. The final string is an encoding of some object, often fractal; the string is interpreted into geometry by mapping each symbol in the string to some geometric primitive or action. (A popular mapping is to have the symbols represent "turtle movement": move forward, move backward, turn left, turn right).

Basic L-systems do not consider information about the surroundings, since the interpretation into geometry happens at the end of the process. While basic DLA alters the contents of the grid, other environment variables are not used in DLA simulation either. For modeling interactions with the environment, open L-systems [12] were devised. While the previous environmentally-sensitive L-systems [13] introduced query symbols allowing information about the environment to influence development, open L-systems have a symbol in the grammar for two-way communication between the L-system and the environment. The similar notion of open DLA has also been employed for lichen simulation [4]. The main drawback to L-systems lies in the difficulty of devising the system of replacement rules; the connection between the rules and the resulting shapes can be profoundly obscure.

3 Algorithm

Our algorithm involves finding a collection of paths through a weighted graph. The graph is a regular lattice filling the 2D or 3D space where the modeled object is to exist; weights on the edges are chosen at random. To create a dendritic form, we connect together multiple paths which share one endpoint, the root of the dendrite. Our implementation performs a best-first computation of path costs from the root to all nodes in the lattice. The dendritic shape can either be converted to geometry, directly (as lines) or by taking an isosurface from a scalar field; or, the shape can be visualized without the intermediate geometry, in the case of 2D dendrites.

The method for creating dendrites operates as follows. A regular lattice is created, and the edges of the lattice given weights from some distribution. We use four-connected lattices (six-connected in 3D) to conserve memory, but eight-connectivity (26-connectivity) could be used to reduce lattice artifacts; in this case non-orthogonal edges' weights would need to be scaled appropriately. We have found that a uniform distribution of weights, say $W = 1 + r\langle R \rangle$, works well. In the preceding, we denote by W a weight, and let $\langle R \rangle$ be a value chosen randomly from the interval $(0,1)$; r is a parameter determining the amount of fluctuation permitted in the weights. We found a value of r around 10 to work well. Note that with small r the resulting paths are close to Manhattan paths (since the constant term dominates), while with larger r the paths are more erratic (since the random component is relatively more important).

We often perform best-first computation from a more elaborate set of nodes than just a single root node, and we need some terminology to refer to the base nodes (those at distance zero); borrowing from the implicit surfaces terminology, we call this set of nodes the *generators*. The next stage in our algorithm is to choose the generators for the dendrite. If the generators are single disjoint nodes, each one will become the root of a separate dendritic shape, but we commonly choose connected sets of points, as described shortly. Using best-first search, we populate all nodes in the lattice with the costs of their least-cost paths from the generators.

Next, we select a set of endpoints in the lattice. The endpoints can be chosen randomly, determined procedurally, or placed manually. In the examples shown in this paper we placed endpoints almost randomly; we used rejection sampling to prevent two endpoints from appearing too near to one another.

With the endpoints chosen and the graph populated with distance values, we use a greedy algorithm to find the least-cost path from each endpoint to the nearest generator. The union of the paths thus obtained is the dendrite. The overall construction process is shown in Figure 2.

3.1 Path Refinement

Our method as described so far produces shapes with resolution limited by the fixed resolution of the mesh. However, there is a natural extension to an iterative refinement approach: once the skeleton of the dendrite has been created, or the shell of the object in the case of a mesh from a segmentation, a new higher-resolution lattice can be constructed in the region of interest. This refinement process can be repeated if desired. The basic idea is that a new sublattice is built for each node in the dendrite; the sublattices

Fig. 2. Plain dendrite construction process. Left: isocontours of distance values from central generator. Middle: randomly placed endpoints. Right: dendrite arising from planning paths from endpoints to central generator.

Input: a coarse path D consisting of m nodes.
Output: a refined path D'.

1. For each node in D, say N_i, create a regular lattice L_i of size $n \times n$. Assign positions to nodes in L_i relative to N_i.
2. For $i = 0$ to $m-2$, stitch the lattices together by adding edges between nodes in L_i and L_{i+1}. Call the resulting graph G.
3. Perform a path planning task within G and return the result.

Fig. 3. Pseudocode for refining a path

are hooked together to form a connected graph, where a new pathfinding process can take place. The left part of Figure 4 illustrates the process.

Pseudocode describing the refinement process for a single path is shown in Figure 3; the process is repeated for each path in a dendrite. One advantage of doing the refinement on a per-path basis is that the high-resolution graphs are individually small, and they are temporary, and hence memory usage is not overly onerous. A side-by-side comparison between a coarse dendrite and a refined dendrite is shown in Figure 4. The presence of the lattice is much less visually obvious in the refined version. The refinement can also be applied to 3D lattices to create a high-resolution 3D model.

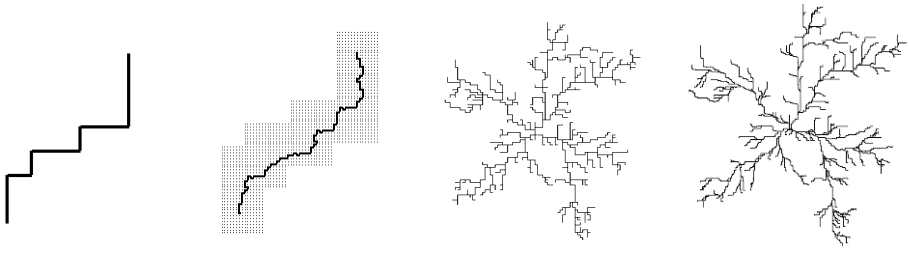


Fig. 4. Left image pair: a coarse path, and a refined path computed inside a finer lattice around the coarse path. Right image pair: a dendrite generated on a coarse graph, and a refined version of the coarse dendrite.

Input: weighted graph G, containing nodes N and edges E; an initial generator Z; parameters α and β; initial number of centres m; initial distance d. Output: a list of nodes P on the fractal dendrite.
1. Set P to null; append Z to P.
2. Repeat the following while $d > \varepsilon$.
2A. Find m centres, at distances $\approx d$ from Z.
2B. Find a path from each centre to Z. Append each path to P.
2C. Set m to $m * \beta$.
2D. Set d to d/α.
2E. Set Z to P.

Fig. 5. Pseudocode for creating fractal dendrites

3.2 Fractal Site Placement

A fractal dendrite can be created through an iterative process involving repeatedly adding new endpoints, and the corresponding paths, to an existing structure. In the first iteration, the structure is a single root node. In later iterations, we compute paths to the entire structure obtained at the previous iteration.

At each iteration, we increase the number of endpoints to be placed by a branching factor β. At the same time, the maximum distance from the generators that each new endpoint is placed is reduced – divided by a factor α, the attenuation factor. The process continues until the maximum distance is less than some small value, say 2 pixels. Notice that the number of iterations therefore depends on the attenuation factor; a larger factor means that the maximum distance decays to a value beneath the threshold more rapidly, resulting in a sparser dendrite. Figure 6 gives a visualization of this process; pseudocode describing the process is given in Figure 5. Images showing different fractal dendrites are shown in Figure 7.

We also slightly modify the nature of the generator. In the previous algorithm, the generator was considered to be at distance zero. Here, we compute different distances for different nodes on the generator: the distance of a point is some factor less than one (say 0.5) times its distance as calculated in the previous iteration of best-first search. The effect of introducing this factor is to cause later paths (those computed on a later iteration) to meet earlier paths at an angle, seemingly anticipating the direction of growth of the dendrite. This tendency can most clearly be seen in the early iterations of Figure 6.

3.3 Converting to Geometry

In this paper, we employ one of two options for converting the description of path locations to geometry. One option is to use the paths nearly directly, and render each node on the path as a simple geometric object, e.g., a sphere. This approach has the advantage of extreme simplicity.

Another option is to create a distance field from the structure and extract an isosurface from the field. The distance field can be computed using the machinery we already have in place: using the dendrite as the generator, we make a pass of best-first search over the

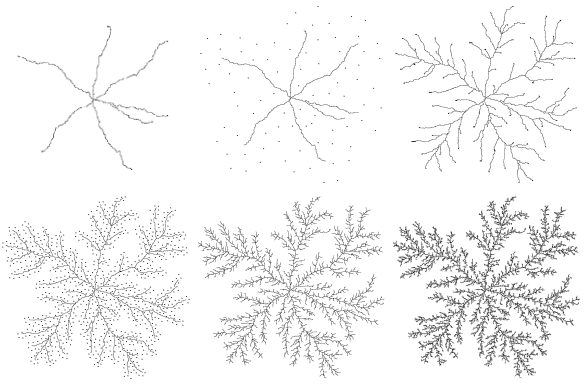


Fig. 6. A fractal dendrite (four iterations). Initially, we have only a few branches, but successively more endpoints are placed at successively smaller distance from the structure.

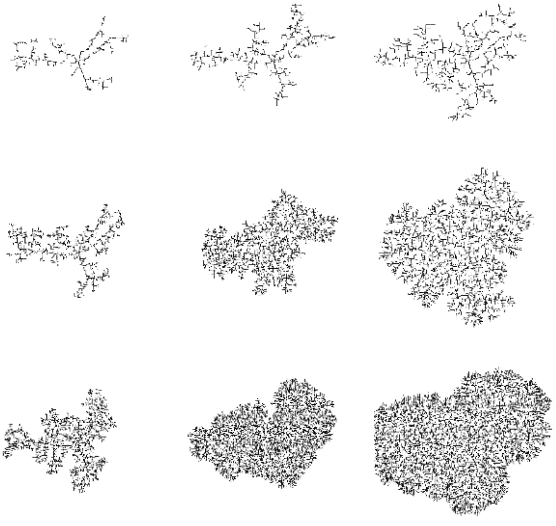


Fig. 7. A few fractal dendrites, with different parameters governing the branching factor and distance limit at each iteration. Right to left: $\alpha = 2, 1.5, 1.2$; top to bottom, $\beta = 2, 3, 4$. All dendrites were built with three iterations.

lattice. The best-first search visits each node in the lattice, populating it with the least-cost path distance from the generators. The resulting distance field can then be converted to geometry using an existing isosurface extraction algorithm such as marching cubes. A visualization of the 2D isosurfaces from a dendritic generator are shown in Figure 8.

A third option, not shown, is to interpret the scalar field of distance values as a height field or displacement field. With this option, the lattice becomes the mesh; the x and y coordinates of a mesh vertex come from the 2D location of the lattice node, and the z coordinate comes from the distance value stored in the node. This approach might be suitable for terrain synthesis, for example; approximating a general terrain by

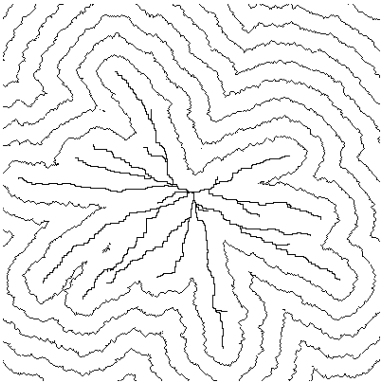


Fig. 8. Two-dimensional isocontours from a dendritic generator. A surface can be generated by taking one of these contours.

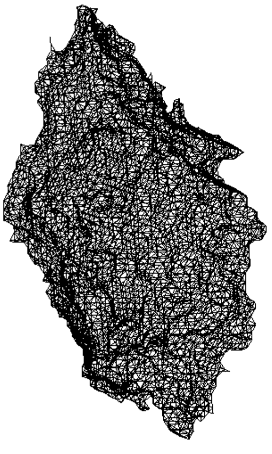


Fig. 9. "Rock" mesh from lattice segmentation

a height field is common in computer graphics. Venation patterns in leaves might also be imitated by treating the distance values as heights.

3.4 Creating Solid Objects with Region Marking

The same workflow used to create dendritic shapes can also be used to generate irregular solid objects. Specifying multiple disjoint nodes as generators, and keeping track during the best-first search process of which site is nearest a given lattice node, has the effect of segmenting the volume. One region is created for each generator node, consisting of all the points nearest that node. A mesh marking the boundary of one of these regions is shown in Figure 9.

The regions from the segmentation are locally irregular but have a simple overall shape. We have referred to them as "rocks" because they resemble broken pieces of some hard and not necessarily homogeneous material.

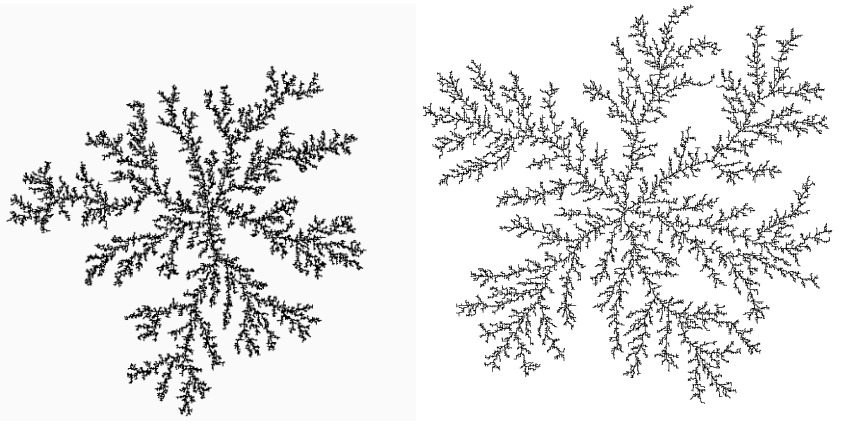


Fig. 10. Left: dendritic form generated by diffusion-limited aggregation. Right: imitation of DLA with a path planned fractal (4 iterations).

4 Results and Discussion

We next show some further results created by our method, in the form of images and models. We have already shown several examples of simple dendritic forms, from Figure 1 onward. In this section we give more elaborate models and provide some commentary on the types of models that our approach can generate. We give unadorned skeletal models and meshes; more sophisticated rendering, including texture mapping, could improve the final images, but in this paper we are focusing on the models themselves.

We can readily create dendritic forms, i.e., branching structures. Branching comes about in our model owing to the use of a common graph for all pathing queries: paths to nearby destinations will often share the early portion of their route, so that a single path appears to emerge from the source, branching when the two previously overlapping paths deviate. The same reasoning, plus the fact that we use a common set of source nodes for all paths, means that the paths will never cross one another. Given a consistent tie-breaking mechanism, there is a unique path from the source nodes to any node in the graph; hence, two paths that meet do not cross, but rather share the same path the rest of the way to the generators.

We have set out to imitate the dendritic forms of DLA, and the results of this imitation are shown in Figure 10. The two models are visually extremely similar, although a detailed investigation would reveal the limited nature of the path planned dendrite (it is fractal only over the small range of scales explicitly programmed in). However, to the unaided human eye the structures look extremely similar; the pragmatic difference is that the path planned dendrite took about 100 times less computer time to create.

We used our system to build a model of staghorn coral, shown in Figure 11. The coral model was created by manually placing endpoints in a 3D graph; the points were not chosen to exactly duplicate the input model, but to give a visually similar appearance, i.e., the synthetic coral could plausibly have come from the same underlying growth

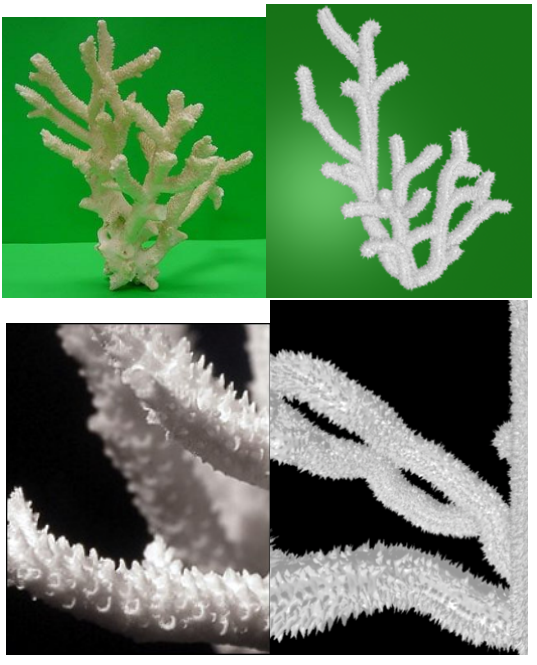


Fig. 11. Left: real coral. Right: coral generated using path planning.

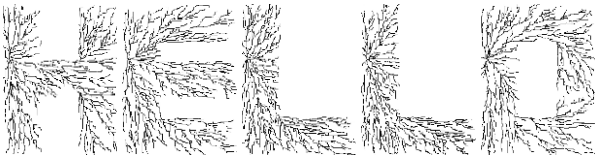


Fig. 12. "Hello" written with dendrites

process. Despite the small amount of information provided to the modeler (only the endpoints of the branches were specified), the synthetic coral model resembles the real coral quite well.

The synthetic image was rendered using Pixie (pixie.sourceforge.net), with the high-frequency structure (thorns) on the surface of the branches obtained from a Renderman displacement shader.

Figure 12 demonstrates one way to exploit the graph to give high-level control over the dendritic shapes. In generating this figure, we created a separate graph for each letter, and arranged the nodes of the graph into the shape of the desired letter. The resulting paths filled a portion of the space within the graph, causing the letter to become visible. A similar mechanism could be used to generate three-dimensional forms, in a manner akin to the synthetic topiaries of Prusinkiewicz et al. [13]. Our 2D result is comparable to the lichen-writing of Desbenoit et al., who distributed seeds for DLA in letter-shaped regions.

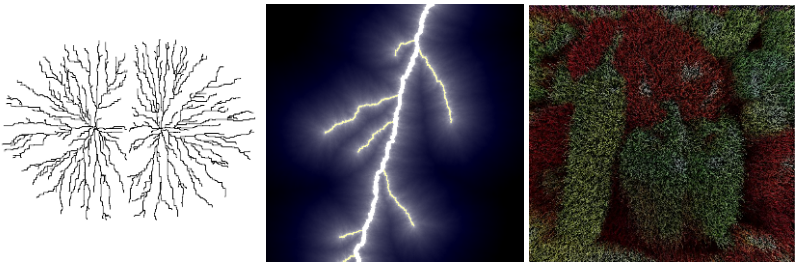


Fig. 13. Left: competition for space on the part of two lichens. Middle: endpoint placement producing a shape resembling lightning. Right: a mossy version of the peppers image.

Competition for space is one of the phenomena simulatable within the framework of open L-systems. We can imitate this phenomenon within our framework by placing multiple disjoint generators within our graph and scattering endpoints nearby. An example of competition is shown in the left of Figure 13. Although the dendrites do not actually communicate during the path planning process, the dendritic forms appear to exhibit an avoidance behaviour.

A dendritic form akin to lightning is shown in the middle of Figure 13. For this 2D example, little needs to be changed from the typical 2D dendrites we previously showed: the generator is a single node at the top of the image, and we initially placed a small number of endpoints at the bottom of the image, producing the main lightning strokes. Subsequently, we added more endpoints in the vicinity of the main strokes, with a bias towards placing them lower in the image. This approach can straightforwardly be extended to 3D by using a 3D lattice.

The lightning example illustrates another control mechanism present in our framework: the endpoints of the dendrite can be specified exactly. For example, lightning might be made to strike a specified "lightning rod" location. In a computer game context, there might be a lightning weapon with user-specified targets. DLA can offer

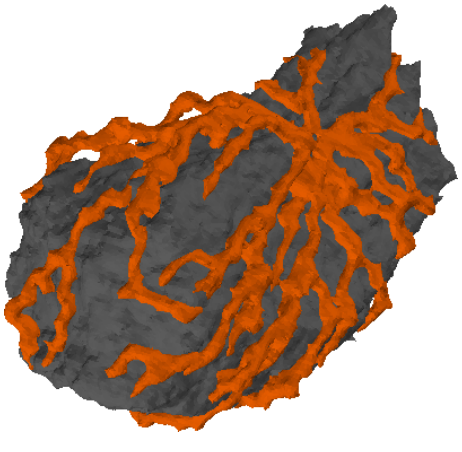


Fig. 14. Lichen on rock, both generated with our method

Table 1. Table of model timing results

Model	lattice	endpoints	time
Simple dendrite	600^2	15	0.94 s
Fractal dendrite	512^2	8930	7.55 s
Coral (no refinement)	50^3	11	1.06 s
Coral (with refinement)	50^3	24	3.06 s

control over the endpoints of the main branches (by starting random walkers from the location the branch should reach), but control over positions of secondary branches is more difficult to achieve.

We can produce forms resembling moss by having a plane or other surface as a generator, and computing paths to destinations near the surface. The right part of Figure 13 shows a large number of paths computed up from the plane. In this example, paths are rendered as chains of line segments with lighting given by the variant of deep shadow maps [11] proposed by Bertails et al. [2], and the colors of the paths taken from the peppers test image. Hair or fur could be generated similarly.

Figure 14 shows a lichen growing over a rock. Both the lichen and the rock models were generated using our method: the rock mesh by segmenting a lattice as described in section 3.4, and the lichen by placing both the path endpoints and the dendrite generator on the rock and forbidding paths to enter interior rock nodes. Notice, therefore, that the modelled lichen is able to leave the surface of the rock briefly before returning; had we so chosen, we could easily have constrained the lichen to the rock face, simply by considering only nodes on the rock boundary. The rock and lichen models in Figure 14 demonstrate the flexibility of our approach, with two quite different models generated by the same underlying process.

Table 1 shows timing results for our method. The figures in Table 1 give times for model construction; these models were either shown directly or were rendered using spheres. Timing results are given for a 1.8GHz P4 with 512 MB RAM. When isosurface extraction is used to create a mesh for rendering, the marching cubes algorithm can take an additional three to seven seconds depending on the complexity of the resulting mesh (creating a surface from a 128^3 distance field).

For 2D dendrites, the sub-one second modeling time can be considered interactive, so that different parameter settings can be experimented with live. The 3D modeling times, albeit on a somewhat coarser grid, are nonetheless only around 10 seconds; for comparison, the lightning simulations of Kim and Lin require hours, and the ice simulations (in 2D) still require at least a few minutes. Desbenoit et al. [4] give times ranging from 1 second to nearly 500 seconds, depending on the complexity of the generated lichen. The DLA image shown in Figure 10 was generated at a resolution of 500×500 with 25000 particles; the basic random walker algorithm was used on a 3.2GHz P4 and required about 7.5 minutes to complete.

5 Conclusions

We have presented a fast, simple method for generating dendritic forms. Because path planning has been well studied in computer science, many standard algorithms exist and

should be familiar to computer graphics practitioners; in consequence, our algorithm is easy to implement. The path planning formulation creates dendrites extremely quickly: less than a second for simple structures, and less than 10 seconds for complex fractal and 3D structures. Orders of magnitude more time are required for DLA and other reported systems for creating dendritic shapes.

The range of natural objects expressible as dendritic forms is great. In addition to dendrites, the path planning approach can generate irregular solid objects by segmenting an input mesh. The versatility of dendrites, combined with the ability to generate irregular solid models, gives our method potentially wide applicability. Unlike L-systems, the path planning framework is not very mature, and much remains to be discovered. For example, future work can address the endpoint placement process, perhaps by distributing them procedurally in a more sophisticated way. We can also plan paths within a general graph, rather than always using regular lattices; removing this restriction could produce even more natural structures without enormously increasing our node budget.

In this paper, we have given a broad overview of the dendritic shapes our method can generate. One avenue for future work is to narrow in on specific phenomena and provide specific advice and parameters for generating each particular phenomenon. Lightning, trees, lichens, and terrains have long been of interest in computer graphics, and we find moss a particularly intriguing direction. Other kinds of structures, such as cracking, venation, and crystals, are also possible.

We would like to investigate ways of achieving dynamic dendrites within our framework, since animations of growing dendrites are sometimes desired. One possibility is to compute the completed dendrite, and use the costs associated with the path nodes to determine the time at which a given node should be added to the growing dendrite.

Acknowledgements. Thanks to the IMG lab at the University of Saskatchewan for helpful suggestions. Particular thanks to Peter O'Donovan for the mossy peppers image. This work was supported by NSERC RGPIN 299070-04.

References

1. Ball, P.: The Self-Made Tapestry: Pattern Formation in Nature. Oxford University Press, Oxford (2004)
2. Bertails, F., Ménier, C., Cani, M.-P.: A Practical Self-Shadowing Algorithm for Interactive Hair Animations. In: Proceedings of Graphics Interface 2005, pp. 71–78 (2005)
3. Bunde, A., Havlin, S.: Fractals and Disordered Systems. Springer, Heidelberg (1996)
4. Desboinet, B., Galin, E., Akkoche, S.: Simulating and Modeling Lichen Growth. Computer Graphics Forum 23(3), 341–350 (2004)
5. Ebert, D., Musgrave, F., Peachey, D., Perlin, K., Worley, S.: Texturing and Modeling: A Procedural Approach, 3rd edn. Morgan Kaufmann, San Francisco (2003)
6. Gastner, M., Newman, M.: Shape and efficiency in spatial distribution networks. Journal of Statistical Mechanics 1(P01015) (2006)
7. Kim, T., Henson, M., Lin, M.: A hybrid algorithm for modeling ice formation. In: The 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 305–314 (2004)
8. Kim, T., Lin, M.: Visual simulation of ice crystal growth. In: The 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 86–97 (2003)

9. Kim, T., Lin, M.: Physically based animation and rendering of lightning. In: Pacific Conference on Computer Graphics and Applications 2004, pp. 267–275 (2004)
10. Lindenmeyer, A., Prusinkiewicz, P.: The Algorithmic Beauty of Plants. Springer, Heidelberg (1990)
11. Lokovic, T., Veach, E.: Deep shadow maps. In: Proceedings of SIGGRAPH 2000, pp. 385–392 (2000)
12. Mech, R., Prusinkiewicz, P.: Visual models of plants interacting with their environment. In: Proceedings of SIGGRAPH 1996, pp. 397–410 (1996)
13. Prusinkiewicz, P., James, M., Mech, R.: Synthetic Topiary. In: Proceedings of SIGGRAPH 1994, pp. 351–358 (1994)
14. Winston, P.: Artificial Intelligence. Addison-Wesley, Reading (1992)
15. Witten, T., Sander, L.: Diffusion-Limited Aggregation, a Kinetic Critical Phenomenon. Physical Review Letters 47(19), 1400–1403 (1981)

The Orthant Neighborhood Graph: A Decentralized Spatial Data Structure for Dynamic Point Sets

Tobias Germer and Thomas Strothotte

Department of Simulation and Graphics, Otto-von-Guericke University of Magdeburg
Universitätsplatz 2, 39106 Magdeburg, Germany
{germer,tstr}@isg.cs.uni-magdeburg.de

Abstract. This work presents a novel approach for proximity queries in dynamic point sets, a common problem in computer graphics. We introduce the notion of Orthant Neighborhood Graphs, yielding a simple, decentralized spatial data structure based on weak spanners. We present efficient algorithms for dynamic insertions, deletions and movements of points, as well as range searching and other proximity queries. All our algorithms work in the *local neighborhood* of given points and are therefore independent of the global point set. This makes ONGs scalable to large point sets, where the total number of points does not influence local operations.

Keywords: Dynamic point sets, proximity queries, range searching, geometric spanners, particle systems.

1 Introduction

In computer graphics, many methods rely on dynamic point sets. One example are particle systems, where individual particles can be considered as points moving through space [1]. A more complex example are multi-agent systems, where each object has some complex behavior [2]. A common task in these systems is to find all neighbors in a defined neighborhood or the nearest neighbor for a particle or agent. This paper introduces a novel method to efficiently handle such *proximity queries* in dynamic point sets.

Our goal is to develop a simple and efficient data structure that maintains dynamic point sets. It should provide fast access to the local neighborhood for each point. Moreover, it should support big point sets commonly occurring in particle or agent systems. Popular approaches for this problem include (hierarchical) space partitioning techniques like octrees or bucket grids. However, these methods are either inflexible, do not perform well in dynamic settings, or do not scale well for large point sets.

We present an alternative paradigm which provides a flexible, decentralized approach for proximity queries in dynamic point sets. The main idea is to use a very simple, graph-based data structure with low memory footprint. We provide efficient algorithms which act in the local neighborhood of the points. This makes them input and output sensitive, as well as scalable to large point sets. Therefore, local changes in the point

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 41–55, 2008.

configuration only result in local changes in the data structure. Local operations like searching all neighbors in a given radius or moving a point a small (local) distance do not depend on the total size of the point set. This makes our approach suitable for dynamic particle or agent systems, where typical movements are relatively small and local.

Our approach is novel and poses many unresolved questions. The goal of this paper is to introduce the basic ideas and to describe the principles of our algorithms. We leave a thorough analysis as well as an evaluation and the application of our approach for future work. Finally, we restrict our problem in this paper to the 2D case, i.e., to planar point sets.

2 Background

Tasks like locating points, finding their neighbors or maintaining dynamic point sets are a common problem in computer graphics and computational geometry. Numerous approaches have been introduced and analyzed, making spatial data structures an elaborate area of research. We restrict our treatment of related work to the most established techniques used in computer graphics.

Uniform Space Subdivision. A simple way to speed up proximity queries [2] or collision detection [3] is to divide the space into equally sized buckets where the objects are stored. Objects within a given range are found by considering only the intersected grid cells. Although simple and often effective, this technique has a relatively large memory footprint and doesn't work well in settings with varying search radius or inhomogeneous point distributions.

Spatial Hashing. Instead of storing the grid explicitly, spatial hasing employs a hashing function based on the grid cells to store objects in a hash table. This way, sparse and possibly infinite scenes can be managed [4]. However, the size of the grid cells still depend on the expected search radius, so that the technique becomes inefficient for range searching with varying search radius.

Quad- and Octrees. A widely used technique to overcome these problems is to adaptively subdivide space by quad-trees (resp. octrees). An overview of different variations gives Samet [5].

BSP- and kd-Trees are another class of popular techniques, supporting very flexible space subdivision and also working in higher dimensions [6].

Bounding Volume Hierarchies. Instead of subdividing space, bounding volume hierarchies like OBB-Trees [7] or BD-Trees [8] approximate the input data hierarchically to accelerate collision detection, for example.

Although providing fast (logarithmic) access to arbitrary leaf objects, all these tree-based approaches have a global (centralized) structure and therefore depend on the total number and structure of points. Instead, we seek for a data structure where local operations do not depend on the global structure. In general, tree-based approaches have also difficulties maintaining dynamic objects like moving point sets. Often the

trees become inefficient or large parts have to be rebuild after a series of insertions or deletions.

3 ONGs

The main inspiration for our approach are the principles of swarm behavior and swarm intelligence [9]. In biology, there are various examples of large groups of animals like flocks of birds or schools of fish which rapidly move in global formations without collisions [10]. However, a single animal has neither the mental nor physical ability to track all other animals and maintain a global view on the swarm as it steers through space. Instead, every animal only knows its *local neighborhood*, i.e., nearby animals and the local environment. Every animal acts solely based on this local information. However, the whole swarm is connected through various neighborhoods. This way, global information can be distributed using local structures. Therefore, global patterns can emerge.

We adopt this principle to build a data structure for point sets where each point tracks a limited, *local* neighborhood. This results in a decentralized data structure which provides local information. In addition, each point must have access to arbitrary large (global) neighborhoods, if needed. This way, every point *indirectly* knows the total point set. Thus, we have two main requirement:

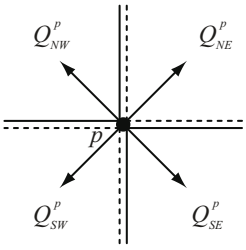
- each point has a constant number of neighbors
- each point must have access to every other point

To meet the first requirement, we store pointers to all local neighbors for each point. The result is a directed geometric graph, where the vertices correspond to the points of the point set, and arcs represent the neighborhood relationships.

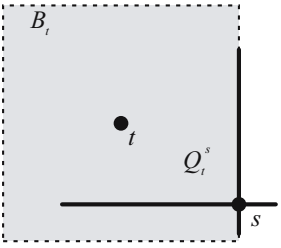
The second requirement implies that the graph has to be strongly connected. There must be a path (i.e., a chain of neighbors) connecting each vertex with every other vertex. To meet this requirement, we have to consider which vertices exactly are neighbors and how many neighbors are required for each vertex.

To answer this question we use results about "t-spanners" and "weak spanners" from Fischer et al. [11,12,13]. We first review the relevant concepts. Geometric spanners are important data structures in computational geometry, because they approximate the complete graph using only $O(n)$ edges, where n denotes the number of vertices [14]. In our context, this means that we can approximate global information about the point set with local neighborhood relationships. A geometric graph $G = (V, E)$ is called *t-spanner*, if for each pair of vertices $(u, v) \in V$ there exists a path in E, which is no longer than t times the direct distance between u and v. Thus, the (relative) length of the path of any pair of vertices is bounded by the stretch factor t. The complete graph is obviously a t-spanner with $t = 1$. However, it has an out-degree of $n - 1$ and therefore takes $O(n^2)$ space. Instead, we need a low and constant out-degree for our data structure to be output sensitive.

One way to construct a t-spanner is to divide the space around each vertex p into k cones and to create a directed edge from p to the closest vertex in each cone [15]. It can be proved that the resulting graph is a t-spanner for $k > 6$ cones [16]. Fischer et al. improve this value to $k \geq 4$ by introducing "weak spanners". However, these graphs



(a) p links to the nearest neighbor for each quadrant Q. The coordinate axes are assigned to Q^p_{NW} respectively Q^p_{SE}.



(b) t is located in quadrant Q^s_t of s. The box B_t contains all points nearer to t than s according the the maximum-metric.

Fig. 1. Basic construction of ONGs

only satisfy a weak spanner property. Here, not the path length between any vertex pair is bounded, but the (Euclidean) distance from any vertex on the path to the start vertex. Note that this graph must be strongly connected. In the prove of this property for $k \geq 4$, Fischer et al. also show a way how to actually find a short path between any two vertices.

We use the weak spanner construction from Fischer et al. in a slightly adapted version. We divide the (planar) space around each point p into the four quadrants $Q^p_j, j \in \{NE, NW, SE, SW\}$ defined by the coordinate axes. We have to take care about the coordinate axes themselves and assign them to unique quadrants. Fischer et al. introduced a consistent scheme for this, as illustrated in figure 1(a). We assume that there are no coincident vertices. We use the Manhattan-metric $d_M(p, v) = |p_x - v_x| + |p_y - v_y|$ to find the nearest points $v_j \in Q^p_j$. The motivation to use the Manhattan-metric is explained in section 4.5. Finally, we store each v_j as a local neighbor for p. Figure 1(a) illustrates this concept. The resulting structure also generalizes to higher dimensions. Therefore, we call it "Orthant Neighborhood Graph" (ONG)[1]. This graph has appealing properties:

- *Constant Outdegree.* Each vertex has at most four local neighbors. A graph with n vertices has at most $4n$ edges.
- *Quadrant-based Partition.* By aligning the cones with the four quadrants we can easily assign points to cones using coordinate comparisons. Employing only simple comparisons of constant numbers also makes our approach robust.
- *Simple Metric.* The Manhattan-metric is simple and cheap to compute.
- *Weak Spanner.* The resulting graph is strongly connected and has the weak spanner property.

To verify that ONGs are strongly connected, we briefly sketch the main argument from the prove presented by Fischer et al. [12]. Consider two vertices s and t as illustrated in figure 1(b). To construct a path from s to t, we consider the quadrant Q^s_t of s, to which t belongs. By definition, s must have a neighbor n for this quadrant. If this

[1] The concept of quadrants and octants generalized to arbitrary dimensions is called "Orthant".

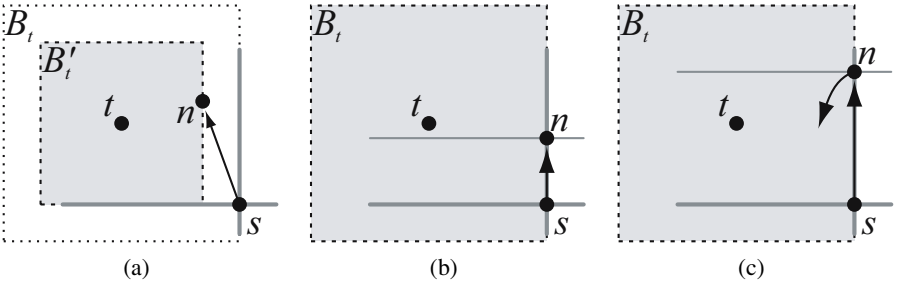


Fig. 2. Cases for the location of neighbor n for vertex s

neighbor is t, we are done. Otherwise, there must be a neighboring vertex n closer to s than t. We show, that by recursively following the neighbor n, we incrementally get closer to t, until we reach t. Let $B_t = \{x \in \mathbb{R}^2 : d_{max}(t,x) \leq d_{max}(t,s)\}$ be the square defined by the maximum-metric $d_{max}(u,v) = max(|u_x - v_x|, |u_y - v_y|)$ (see figure 1(b)). Then, the neighbor n must be contained in B_t. There are three cases:

1. $d_{max}(t,n) < d_{max}(t,s)$: The neighbor is inside B_t and therefore closer to t. (See figure 2(a).)
2. $d_{max}(t,n) = d_{max}(t,s)$ and $Q_t^s = Q_t^n$: s is on the border of B_t and t is in the same quadrant with respect to n. Then, n is still nearer to t according to the Manhattan-distance. (See figure 2(b).)
3. $d_{max}(t,n) = d_{max}(t,s)$ and $Q_t^s \neq Q_t^n$: s is on the border of B_t and t has changed the quadrant with respect to n. Then, we do not come closer to t and B_t stays the same. However, in the next step of the path, the neighbor of n cannot be longer on the border of B_t, because our assignment of coordinate axes to quadrants as shown in figure 1(a) does not permit this. Therefore, B_t will get smaller in the next step. (See figure 2(c).)

This shows that the square B_t gets smaller or stays the same with each step along the path. However, cases 2 and 3 ensure that B_t can only stay constant for a finite number of steps. Therefore, the path will finally reach t. We conclude this section with the following property of ONGs, which is important for the algorithms presented in the next section:

Corollary 1. *Given a vertex s in an ONG, any vertex t in this ONG can be reached by recursively following the neighbor of the quadrant, in which t is located.*

4 Algorithms

Having introduced the fundamental structure of ONGs in the previous section, we now describe algorithms for the dynamic construction of ONGs. We first present the high

level algorithms, then the low level procedures and finally an efficient algorithm for (localized) range searching using ONGs.

4.1 Insertion

First, we introduce two notations:

- $neigh_q^p$ denotes the neighbor of vertex p saved for quadrant q
- $quadrant_p(s)$ returns the quadrant of p in which vertex s is located

To insert a new vertex in an ONG, we have to insert and change certain arcs of the graph, so that the ONG stays consistent. We use the following algorithm:

Procedure. `Insert(`*Vertex s, Vertex p*`)`

Input: a starting vertex s that is already inserted
Input: the new vertex p
1 **if** *ONG is empty* **then**
2 insert p as the first vertex;
3 **return**
4 **for** *each quadrant q of p* **do**
5 s_q = search some point in q, starting at s;
6 $neigh_q^p$ = nearest neighbor in q, starting at s_q;
7 **if** $neigh_q^p \neq \emptyset$ **then** $s = neigh_q^p$;
8 search all vertices r_i for which p is the new nearest neighbor;
9 **for** *each* r_i **do**
10 Quadrant $q = quadrant_{r_i}(p)$;
11 $neigh_q^{r_i} = p$;

We first check if the ONG is empty. In this case, p is the only vertex and there is nothing to change (lines 1-3). Otherwise, we search the nearest neighbors for each quadrant of p using the algorithm of section 4.4, and save them as neighbors (lines 4-6). By searching the nearest neighbors, we actually *localize* p, i.e., we determine its local neighborhood. Note that we need a given vertex s, where we begin our search for start vertices s_q in every quadrant, which are then used to initialize the nearest neighbor search. If we found a nearest neighbor, we use it as the starting point for the next quadrant, because it is probably close to the nearest neighbor in this quadrant (line 7).

Afterwards, we have to find all vertices r_i in the ONG, which have p as their new nearest neighbor (line 8). Section 4.5 describes an algorithm for this. Finally, we update these vertices and store p as their nearest neighbor in the according quadrant. Note that we first find *all* vertices r_i before changing the topology of the existing graph. If we would change the topology (i.e., store p as the new nearest neighbor) immediately after we found one r_i, the topology wouldn't be consistent anymore, breaking the assumptions for subsequent queries. Therefore, we strictly separate queries using the ONG from changing the ONG.

4.2 Deletion

To delete a vertex p from the ONG, we have to change all arcs pointing to p. We do this with the following algorithm:

Procedure. `Remove` (*Vertex p*)

```
1 search all vertices r_i for which p is a nearest neighbor;
2 for each r_i do
3     q_i = quadrant_{r_i}(p);
4     s_i = search second nearest neighbor in q_i;
5 for each r_i do
6     neigh^{r_i}_{q_i} = s_i;
```

Note that we don't have to localize p this time, because the local neighborhood (i.e., the nearest neighbors) are already known. We first find all vertices r_i, for which p is a nearest neighbor (line 1). This is similar to line 8 of the insert algorithm and also detailed in section 4.5. For all vertices r_i we have to remove p as a neighbor and store the second nearest neighbor instead. Again, we have to separate the queries for the second nearest neighbor (lines 2-4) from the change of topology (lines 5-6). This results in a consistent ONG where no arc points to p anymore. Therefore, p can be removed.

4.3 Movement

Our goal for ONGs was to design a spatial data structure which can be used to maintain particle or agent systems. A typical property of such systems is that the entities (i.e., the vertices) are moving. We handle this action by simply deleting the vertex and re-inserting it at the new position:

Procedure. `Move` (*Vertex p, Vector v*)

```
1 Vertex s = some neighbor from p;
2 Remove (p);
3 move p according to v;
4 Insert (s,p);
```

This algorithm benefits from small movements of vertices. We take some (old) neighbor of vertex p (line 1) as the starting point for the re-insertion in line 4. If the new position of p is relatively close to the old position, the localization step of the insert procedure will be cheap (see next section). This way, the algorithm becomes *input sensitive*: local movements only traverse and change local parts of the ONG. The downside is that chaotic, global movements traverse very large parts of the ONG, degrading performance. However, particle and agent systems mostly exhibit small movements. Therefore, ONGs will be suitable for such systems.

4.4 Neighbor Searching

One of the first steps of the insert algorithm is the localization of the new point p, where we search for the nearest neighbors for each of its quadrants. This section presents a simple and effective algorithm for this.

In contrast to centralized data structures like quad- or kd-trees, ONGs do not provide a mechanism to quickly locate arbitrary points in the point set. Instead, we have to iteratively traverse the local neighborhood of certain vertices until we find the nearest neighbor for p. The idea is to take a starting vertex s and then "walk across" the graph in the direction of p, until no closer vertex can be found anymore. We need two more concepts for this algorithm:

- *Search Regions.* A search region is a rectangular region representing the "undiscovered" space. Only in the search region new results can be found. If the search region is empty, we have found all result points. We can find all vertices in a search region by using corollary 1: given a vertex p, we can find all vertices in the search region by recursively following all neighbors assigned to the quadrants which intersect the search region.
- *Vertex Flags.* To avoid loops when traversing the graph, we mark visited vertices with a flag, denoted as $flag_p$ for vertex p. Before finishing the algorithm, we have to clear the flags again.

We can now formulate our algorithm for finding the nearest neighbor to a point p (which is not yet inserted) in the quadrant according to s, starting at s:

Function. `NearestNeigh(`*Point p, Vertex s*`)`

```
   Output: nearest Vertex to p in the quadrant of s
 1 Quadrant q = quadrant_p(s);
 2 float d = d_M(p, s) ;                                   // init distance
 3 repeat
 4     SearchRegion R = (square centered at p with side length 2d) ∩ q;
 5     set flag_s;
 6     for each quadrant q_i ≠ q do
 7         Vertex n = neigh^s_{q_i};
 8         if nnInternal (n) then break;                          // for
 9     clear all flags;
10 until no nearer neighbor found ;
11 return s

12 Local Function nnInternal ( Vertex n ):
13   if n = ∅ ∨ flag_n is set then return false
14   Quadrant q_n = quadrant_p(n);
15   if q_n = q ∧ d_M(p, n) < d then
16       s = n;                                    // new nearest vertex
17       d = d_M(p, n);
18       return true
19   set flag_n;
20   for each quadrant q_i ≠ q_n do
21       if R cuts q_i ∧ nnInternal (neigh^n_{q_i}) then
22           return true
23 return false
```

First, we save the quadrant q, for which we search the nearest neighbor (line 1). Then, we save the distance d of the nearest neighbor yet found (which is s, at the beginning the

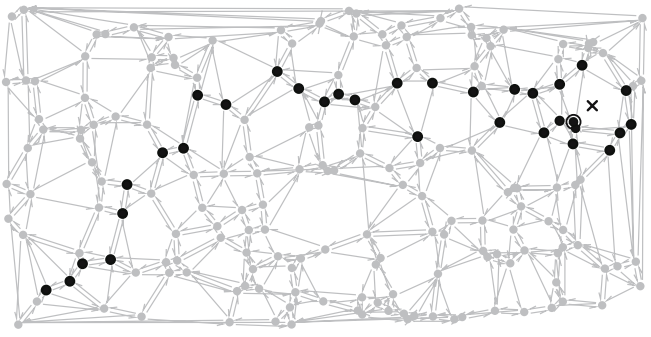


Fig. 3. The black dots highlight the points visited during a nearest neighbor search. The path between the start point in the lower left and the target point in the upper right (cross) is roughly linear.

starting point). Afterwards, we set up a search region R, where all possible nearer neighbors to p could be found (line 4). Note that $R \supset \{x : x \in q \wedge d_M(x,p) \leq d_M(s,p)\}$. We then search for nearer neighbors by calling `nnInternal` for each quadrant q_i of s. Note that we don't have to search in quadrant q, because all points in this quadrant must have a greater distance to p than s. If `nnInternal` finds a new neighbor, it returns $true$ and we start our search again with the new, reduced search region. If no new nearer neighbor could be found, we are finished and return the last result point s.

After ensuring that the current vertex n is not empty and was not visited yet (line 13), the function `nnInternal` checks if n is in the right quadrant and is nearer to p than the current nearest vertex s (lines 15-17). If this is not the case, it recursively performs a simple depth-first search to find other vertices. Note that we only have to search in quadrants that cut the search region.

The algorithm described above can be improved by a simple heuristic. Given the current vertex s (or n in `nnInternal`), chances are high that a new nearest neighbor can be found by following the opposite quadrant of q_s (respectively q_n), because in this quadrant the target point p is located. Therefore, we first search in these quadrants before searching in the remaining ones. This way, we quickly reduce the search region and follow a roughly linear path to the target point p (see figure 3). If the initial starting point s is already close to the nearest neighbor, this path will be very short and only points in a local neighborhood of s resp. p will be traversed. Thus, our algorithm benefits from small movements and local insertions.

In line 4 of the remove algorithm (section 4.2), we search for the *second* nearest neighbor in a given quadrant. This can be done with a slightly adapted version of `nnInternal`. We only have to extend the condition in line 15 to neither accept p nor $neigh_q^p$ as a new nearest vertex. This way, the search is aborted after the second nearest neighbor has been found.

We also use a similar algorithm to find (arbitrary) points in a given quadrant, as required in line 5 of the insert algorithm. In this case, we set the search region to match the quadrant and use a search similar to `nnInternal` to find a point in this region.

4.5 Reverse Neighbor Searching

One step of the insert algorithm of section 4.1 is to search for all vertices r_i, for which a point p is the new nearest neighbor. We say that the vertices r_i *will reference* to p after the insertion. We use a similar step in line 1 of the remove algorithm (section 4.2), where we search all vertices which *are referencing* to a given vertex p. In general, this problem is called "reverse nearest neighbor searching" [17]. Here, we adapt the problem to report all vertices r_i, which have p as their nearest neighbor in *one* of their quadrants.

In the context of ONGs, the number of such vertices can be arbitrary large and depends on the vertex distribution in the local neighborhood around p (consider a vertex surrounded by a circle of other vertices). However, the average number of referencing vertices for each vertex in an ONG is (at most) four, since the out-degree of every vertex is (at most) four.

The idea for our reverse nearest neighbor algorithm is to use a breadth-first search (BFS) constrained by corollary 1 and the two following observations:

Corollary 2. *Let s be a vertex located in quadrant q_s of vertex p. Then, every vertex in the same quadrant q_s of s must be closer to s than to p (see figure 4(a)). Therefore, no vertex in this area can reference to p.*

Corollary 3. *Let s and p be two vertices and $v = s - p$ the difference between them. Without loss of generality, we assume that s is in the upper-right quadrant of p. If $|v_x| < |v_y|$, then all vertices in the upper-left quadrant of p above s are closer to s than to t. Analogously, if $|v_x| > |v_y|$, then all vertices in the lower-right quadrant of p on the right of s are closer to s than to t.*

Fig. 4. p cannot be the nearest neighbor for vertices in the shaded areas. The bold dashed line in (b) marks the Voronoi-edge between s and p for the Manhattan-metric.

This observation can be easily verified by considering the Voronoi diagram for the Manhattan-metric, as illustrated in figure 4(b), where all points above s are also above the Voronoi edge. Note that this observation is only possible with the Manhattan-metric, which was the main reason to use it for ONGs.

We can now present our reverse nearest neighbor algorithm (`ReverseNN`), reporting all vertices which have p as their nearest neighbor.

Procedure. `ReverseNN` (*Vertex p*)

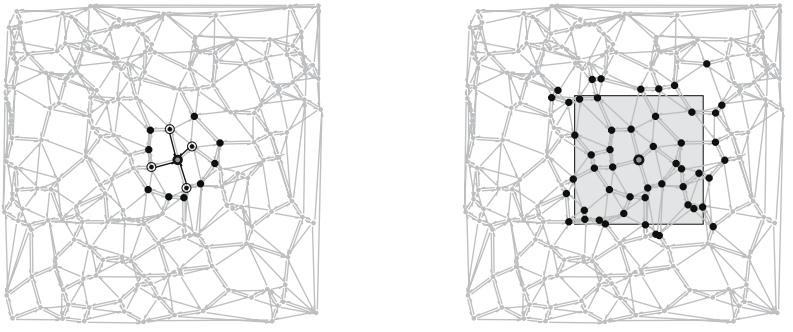
```
1  List Q = empty vertex list;
2  SearchRegions R[4] = four open rectangles corresponding to the quadrants of p;
3  set flag_p;
4  for each quadrant q_i do
5      Vertex n = neigh^p_{q_i};
6      if n ≠ ∅ then
7          set flag_n;
8          clipSearchReg (n);
9          append n to Q;
10 repeat
11     Vertex s = pop front element from Q;
12     Quadrant q_s = quadrant_p(s),   q̂_s = opposite(q_s);
13     if neigh^s_{q̂_s} = p then report s as a result;
14     for each quadrant q_i do
15         Vertex n = neigh^s_{q_i};
16         if (n ≠ ∅) ∧ (flag_n is not set) ∧ (q_i ≠ q_s) ∧ (R[1...4] cuts q_i) then
17             set flag_n;
18             clipSearchReg (n);
19             append n to Q;
20 until Q is empty ;
21 clear all flags;

22 Local Function clipSearchReg ( Vertex n ):
23   Quadrant q_n = quadrant_p(n),   Vector v = n − p;
24   if |v_x| < |v_y| then
25       Quadrant q = q_n inverted in x-direction;
26       clamp R[q] in y-direction;
27   else
28       Quadrant q = q_n inverted in y-direction;
29       clamp R[q] in x-direction;
```

The algorithm begins by creating a search region for every quadrant (line 2). Initially, every search rectangle is equivalent to its quadrant and therefore on two sides open. Then, all neighbors of p are inserted into the BFS queue (lines 4-9). In addition, the search regions are reduced by calling the function `clipSearchReg`, which employs corollary 3 to clip the search rectangles for every neighbor.

Afterwards, we process every vertex s from the queue (lines 10-20): If s has p as its nearest neighbor, it is reported as a result (line 13). Note that p can only be the neighbor for s in the opposite quadrant of q_s. Then, we consider all neighbors of s which have not been visited yet and which correspond to quadrants intersecting one of the search regions ($R[1...4]$). Following corollary 2, we don't have to consider neighbors in q_s (line 16). After clipping the search regions, we include each such neighbor into the BFS queue (lines 17-19). We repeat these steps until the BFS queue is empty and no more referencing vertices could be found. Figure 5(a) illustrates a typical example.

We use a variation of the algorithm above to search all vertices which will reference to a new vertex p after insertion. The only differences are line 5, where we take the nearest neighbors for p found by `NearestNeigh`, and line 13, where we check, if p is nearer than the current neighbor.



(a) (b)

Fig. 5. The black dots highlight the points visited during a reverse nearest neighbor search (a) and range searching (b) for the point in the middle. The shaded square in (b) represents the query region.

4.6 Range Searching

Finally, the main purpose of ONGs is to provide *proximity queries*. The nearest neighbor for any vertex (according to the Manhattan metric) can be simply found by comparing the four neighbors of each quadrant. Another common query in particle or agent systems is to find all points in a given radius. We formulate this as a (circular) range searching problem: given a vertex p, find all vertices nearer than a certain distance r. An overview on this topic gives [18].

We approximate this problem by finding all neighbors in a square with side length $2r$, centered at p:

Procedure. `RangeSearch(`*Vertex p, float r*`)`

```
1  List Q = empty vertex list;
2  SearchRegion R = square centered at p with side length 2r;
3  set flag_p;
4  while p ≠ ∅ do
5      if p ∈ R then report p as a result;
6      for each quadrant q_i do
7          Vertex n = neigh^p_{q_i};
8          if n ≠ ∅ ∧ flag_n is not set ∧ R cuts q_i then
9              set flag_n;
10             append n to Q;
11     p = pop front element from Q;
12 clear all flags;
```

This algorithm implements a simple breadth-first search (BFS), starting at p. Using BFS for range searching with (weak) spanners was introduced by Fischer et al.[12]. They use the (weak) spanner property to constrain the BFS to a certain region around p. In contrast to this approach, we use corollary 1 to constrain the BFS: we only have to continue the search in quadrants which intersect the search region R. This results in

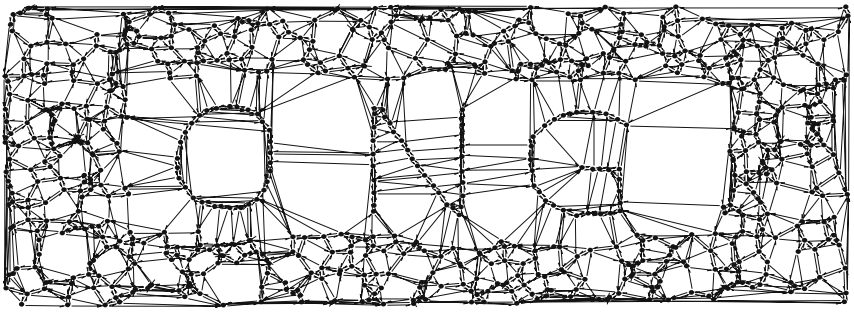


Fig. 6. A complex example of an ONG for a point set with appr. 750 points

much fewer vertices we have to visit. In typical examples, there are only few visited vertices which are not contained in the query rectangle (see figure 5(b)).

Using this algorithm, we can also provide proximity queries for other metrics. For example, the Euclidean nearest neighbor to p can be found by searching all vertices nearer than the nearest neighbor n_M based on the Manhattan metric. We do this by performing a range search around p with radius $d_M(p, n_M)$ and comparing the distances of the result points according to the Euclidean metric.

5 Discussion

In the following we give preliminary results for our approach. We have implemented all presented algorithms and data structures in C++. Figure 6 shows a complex ONG generated with our system.

Table 1 gives timings of some experiments on a 3GHz PC using our prototypical implementation, where the code was not optimized for speed. First, we inserted a large number of points with random distribution in arbitrary order. The table shows that these can be quickly incrementally inserted into an ONG. The incremental removal of all points is even faster, because no localization step (as explained in section 4.4) is needed. If every new point would be inserted nearby a known vertex, the performance would also increase, because the localization step would be cheaper. In the next experiment we compare the cost of locally moving a point in small and large point sets consisting of 1 000, 10 000, and 100 000 points. The timings for all three cases are nearly equal. Finally, we do a range search in small and large point sets. We adapt the radius, so that 100 points are found each time. Again, the timings are nearly equal. Therefore, the total

Table 1. Timings for experiments with ONGs for point sets of different size. The timings in ms are averaged over 10 000 iterations.

Point set	1 000	10 000	100 000
Incremental build	0.031 s	0.367 s	6.141 s
Remove all points	0.027 s	0.309 s	5.694 s
Point movement	0.055 ms	0.059 ms	0.058 ms
Range searching	0.109 ms	0.111 ms	0.110 ms

number of points in the ONG does not influence the cost of local operations like local movements or range searching. This confirms the scalability of ONGs.

6 Conclusions

We have introduced ONGs, a new spatial data structure that supports proximity queries in dynamic point sets. The basis for ONGs are weak spanners, which ensure that storing the nearest neighbor for each quadrant results in a strongly connected graph. ONGs are decentralized in that all information is distributed on the whole point set. We presented different algorithms that work on the local neighborhood of given points. This allows dynamic insertions, deletions and movements of points as well as range queries *independent* of the size of the point set. Our algorithms are input and output sensitive: The cost of moving a point is low for small movements, but grows as it moves farther. Also, the cost of range queries depends of the number and the neighborhood of the result points. These properties make ONGs applicable to systems consisting of a large number of points, like particle or multi-agent systems.

7 Future Work

There are many areas of future work and open questions for ONGs. The approach and the presented algorithms have to be analyzed and evaluated in more detail. A comparison with other techniques (like quad- or kd-trees) will show the usability of ONGs.

We want to adopt ONGs to 3D and higher dimensions. In principle, the presented algorithms and data structures also work in higher dimensions. However, the number of orthants grows exponentially with the number of dimensions, resulting in drawdowns in performance. The lowest known bound of the number of neighbors required for weak spanners in 3D is 8. However, this bound is not tight [13]. ONGs would benefit from a scheme that requires less cones and still produces weak spanners.

Another direction of future work is the kinetization of ONGs. Kinetic data structures store dynamic objects and explicitly model their motion [19]. The idea is to only change the underlying structure if certain predicates change. This could save unnecessary updates of ONGs for small motions which do not change the graph topology.

Finally, we want to apply ONGs to different problems in computer graphics. For example, higher dimensional ONGs could be used for the broad phase of collision detection.

References

1. Reeves, W.T.: Particle Systems – A Technique for Modeling a Class of Fuzzy Objects. In: Computer Graphics Proceedings of ACM SIGGRAPH 1983, vol. 17, pp. 359–376 (1983)
2. Schlechtweg, S., Germer, T., Strothotte, T.: RenderBots—Multi Agent Systems for Direct Image Generation. Computer Graphics Forum 24, 137–148 (2005)
3. Kim, D.J., Guibas, L.J., Shin, S.Y.: Fast collision detection among multiple moving spheres. IEEE Transactions on Visualization and Computer Graphics 4, 230–242 (1998)

4. Teschner, M., Heidelberger, B., Müller, M., Pomerantes, D., Gross, M.H.: Optimized spatial hashing for collision detection of deformable objects. In: Proceedings of the Vision, Modeling, and Visualization Conference 2003 (VMV 2003), Aka GmbH, pp. 47–54 (2003)
5. Samet, H.: The design and analysis of spatial data structures. Addison-Wesley Longman Publishing Co., Inc., Boston (1990)
6. Bentley, J.L.: K-d trees for semidynamic point sets. In: SCG 1990: Proceedings of the 6th Annual Symposium on Computational Geometry, pp. 187–197 (1990)
7. Gottschalk, S., Lin, M.C., Manocha, D.: Obbtree: a hierarchical structure for rapid interference detection. In: Proceedings of ACM SIGGRAPH 1996. Computer Graphics Proceedings, Annual Conference Series, pp. 171–180. ACM Press, New York (1996)
8. James, D.L., Pai, D.K.: BD-Tree: Output-sensitive collision detection for reduced deformable models. ACM Transactions on Graphics 23, 393–398 (2004)
9. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, Oxford (1999)
10. Reynolds, C.W.: Flocks, Herds, and Schools: A Distributed Behavioral Model. In: Computer Graphics (Proceedings of ACM SIGGRAPH 1983), vol. 21, pp. 25–34 (1987)
11. Fischer, M., auf der Heide, F.M., Strothmann, W.B.: Dynamic data structures for realtime management of large geometric scenes. In: Burkard, R.E., Woeginger, G.J. (eds.) ESA 1997. LNCS, vol. 1284, pp. 157–170. Springer, Heidelberg (1997)
12. Fischer, M., Lukovszki, T., Ziegler, M.: Geometric searching in walkthrough animations with weak spanners in real time. In: Bilardi, G., Pietracaprina, A., Italiano, G.F., Pucci, G. (eds.) ESA 1998. LNCS, vol. 1461, pp. 163–174. Springer, Heidelberg (1998)
13. Fischer, M., Lukovszki, T., Ziegler, M.: Partitioned neighborhood spanners of minimal outdegree. In: Proceedings of the 11th Canadian Conference on Computational Geometry (CCCG 1999), pp. 47–50 (1999)
14. Arya, S., Das, G., Mount, D.M., Salowe, J.S., Smid, M.: Euclidean spanners: short, thin, and lanky. In: STOC 1995: Proceedings of the 27th Annual ACM Symposium on Theory of Computing, pp. 489–498 (1995)
15. Yao, A.C.C.: On constructing minimum spanning trees in k-dimensional spaces and related problems. SIAM Journal on Computing 11, 721–736 (1982)
16. Ruppert, J., Seidel, R.: Approximating the d-dimensional complete euclidean graph. In: Proceedings of the 3rd Canadian Conference on Computational Geometry, pp. 207–210 (1991)
17. Maheshwari, A., Vahrenhold, J., Zeh, N.: On reverse nearest neighbor queries. In: Proceedings of the 14th Canadian Conference on Computational Geometry, pp. 128–132 (2002)
18. Agarwal, P.K.: Range searching. In: Goodman, J.E., O'Rourke, J. (eds.) Handbook of Discrete and Computational Geometry, pp. 575–598. CRC Press, Inc., Boca Raton (1997)
19. Basch, J., Guibas, L.J., Hershberger, J.: Data structures for mobile data. In: SODA 1997: Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 747–756 (1997)

Part II
Animation and Simulation

Direct Volume Deformation

Florian Schulze, Katja Bühler, and Markus Hadwiger

VRVis Research Center, 1220 Vienna, Austria
{fschulze,buehler,msh}@vrvis.at
http://medvis.vrvis.at

Abstract. This paper presents an integrated approach for interactive direct volume deformation and simultaneous visualization. The fundamental requirement is that interactive performance without pre-processing must be achieved for large volume data, where at any time up to one million elements participate in a deformation that is applied interactively by picking and dragging in the 3D view. Current physically-based approaches are still one or two orders of magnitude away from this goal. In contrast, our approach extends the non-physical ChainMail algorithm and combines it with on-the-fly resampling and GPU ray-casting. Special transfer functions assign material properties depending on volume density. The affected subvolume is deformed and resampled onto a rectilinear grid on the CPU, and updates the volume on the GPU where it is rendered using ray-casting. While the deformation is already being displayed, its quality is simultaneously refined via an iterative relaxation procedure executed in a parallel thread.

Keywords: Deformation, Resampling, Volumerendering.

1 Introduction

This paper follows a vision first published in 1995: Thought as natural extension to direct volume rendering, Sarah F. Gibson formulated the idea for a system that allows direct deformation, cutting and carving of volume data [6]. She introduced the so called ChainMail algorithm allowing in its extension modeling of deformation of inhomogeneous materials. Similar to direct volume rendering, the deformation is directly performed at the voxel level of the volume without any pre-processing.

The ChainMail algorithm provides only a non physics based deformation scheme, but is able to deform large structures in real time: Having in mind that a small volume dataset of 256^3 consists already of more than 16 million voxels, existing physically based approaches are still far away from being able to deform such structures at interactive frame rates without previous simplification. Due to the limited available computational power at the time of first publication of the algorithm and its extensions, simultaneous volume rendering of the whole dataset during the deformation process was not possible, and Sarah Gibson formulated this task as future work [8].

This paper presents a framework that integrates high quality real time visualization with direct deformation of volume data fulfilling the following requirements:

- Full information of the original data is available throughout the whole process: Deformation and visualization are directly performed at the voxel level of the volume.

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 59–72, 2008.

- The deformation is not physically correct, but plausible depending on the underlying data.
- No time-consuming preprocessing is necessary, like segmentation, simplifications and adaptive hierarchy generation.
- The system reaches interactive frame rates for simultaneous simulation and visualization.

Basis of the proposed deformation system is the Enhanced ChainMail algorithm [16] that is taken as initialization step for a relaxation solver that allows also simulation of elastic deformation. Handling of the high amounts of data has been addressed by a specialized data structure and memory management system. A new image order resampling algorithm has been developed to provide simultaneous visualization of the deformed data using the powerful GPU accelerated volume rendering framework described in [17].

The paper is organized as follows: Related work is discussed in the next section. A short summary of the Chain Mail algorithm and existing extensions is given in section 3. Section 4 outlines the general workflow of our system. The two-step deformation method is explained in section 5 including details on the basic chain mail implementation, relaxation, and material definition. Visualization and related issues are addressed in section 6, interaction methods are discussed in section 7. The paper closes with results in section 8, and a summary in section 9.

2 Related Work

Detailed discussion of the extensively available related work on physically based deformation methods of (volumetric) objects is beyond the scope of this paper. The interested reader is referred to two State of the Art Reports presented at Eurographics 2005 [14,3] giving an excellent general overview.

Considering physically based approaches for direct deformation and visualization of *volume data*, modern point based mesh free methods [12] seem to be the most natural approach to deal directly with medical volume data: theoretically, no preprocessing is required and deformation could be directly performed on the volume if each voxel would be modeled as particle or phyxel.

The approaches mentioned above and reported in [14,3] provide physically correct deformation, but due to their computational complexity, none of them is able to handle more than 100k elements at interactive frame rates, even if GPU accelerated integration schemes are used [9,11]. Simultaneous visualization of deformed objects is another bottleneck, especially if surfaces have to be reconstructed on the fly, like it is the case in general for particle-based and point-based approaches [1]. Nealen et al. [14] stated in the conclusions of the state of the art report: "Yet even with the current methodology, the algorithms and models have seen somewhat limited application in production environments and videos games. One reason for this is the lack of computational power...".

Existing approaches addressing directly the deformation of volumes, i.e. without previous mesh extraction and/or simplification, are mainly based on space or ray deformation techniques: either a coarser structure (e.g. bounding boxes [18], volume or surface geometry [23]) is deformed and the deformation of the volume itself is computed as

displacement based on the deformation of the shape. This can be done either directly or indirectly by deformation of the rays during rendering. But these approaches also do not perform deformation at the the finest level. To capture fine structures, extensive preprocessing (segmentation, geometric reconstruction) has to be done.

Spatial transferfunctions [4] allow geometric, procedural and hierarchical definition of deformations performed on volumes: geometric deformation rules can be assigned to each voxel by a previously defined function. Arbitrary interactive deformation is not possible with this technique.

To our knowledge, the ChainMail algorithm [7] is the only existing algorithm able to perform interactive deformation of common size volume datasets directly on voxel level. The ChainMail algorithm itself is not a physically based deformation method and is only able to simulate plastic deformation, but an additional relaxation step as proposed in [8] can be used to get more realistic and elastic deformations. Furthermore, the connected data structure allows easy manipulation of the volume like cutting and carving. A generalization of ChainMail to arbitrary mesh topologies, the *Generalized ChainMail* Algorithm, has been proposed by [10]. A complete system for planning of arthroscopic knee surgery [5] and a biomechanical simulation of the vitreous humor in the eye [16] based on the (enhanced) chain mail demonstrated the general applicability of the method.

The next section give an overview of the basic functionality and limitations of existing ChainMail implementations.

3 ChainMail Revisited

Data Structures. The classical ChainMail algorithm is designed to operates on data elements initially arranged in a three dimensional axis aligned grid defined by x-,y-,z-axes. Each element is connected with its six direct neighbors by *ChainMail constraints* that are defined as axis aligned regions describing the set of valid positions for each neighbor element. Figure 1 shows the definition of a valid region for a neighbor in x-direction described by its minimal and maximal distance in x-direction and the allowed deviations (shear) in y- and, in the 3D case, also in z-direction. Valid regions for other neighbors are defined in an analogous way. Material properties can be directly modeled by modification of the ChainMail constraints. Extent and form of the valid region are directly connected to stiffness/softness of the simulated material.

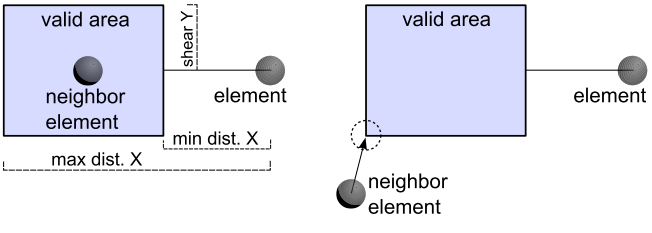


Fig. 1. Left: ChainMail constraints. Right: Constraint violation.

Algorithm. If translation of an element causes constraint violations, i.e. one of the neighbors is moved outside of its valid region, the ChainMail algorithm solves the constraints by sequentially moving the elements into the valid regions. Each moved element can cause new constraint violations, hence the order of element processing is important. The original algorithm provides uniform propagation of the deformation by processing candidates of the six different major directions on a rotational basis.

ChainMail has the advantage that its complexity does not grow with the number of elements of the object but only with the number of affected elements. The performance of the algorithm is based on two features of the algorithm:

1. The deformation is calculated depending on simple constraints.
2. Each element of the dataset is processed at most once per deformation step.

Existing Extensions. The original ChainMail algorithm solves only geometrical constraints. To achieve an optimal energy configuration Gibson presented an additional simple relaxation step in [7]. The algorithm iterates over each element and moves it towards an equilibrium position which is placed in the center of its neighbors. A second drawback of the original ChainMail algorithm is that it is not well suited to process inhomogeneous data. To overcome this limitation the *Enhanced ChainMail* algorithm [16] has been proposed. The equal deformation propagation into each direction is replaced by an importance driven approach where elements with a higher amount of constraint violation are processed first. This method leads to a shock wave like deformation propagation that propagates faster through stiff material. The simple midpoint-based relaxation scheme proposed in connection with the original ChainMail does not allow the definition of material parameters and is therefore not suitable for the enhanced ChainMail. Up to now, no relaxation scheme addressing this problem in connection with the Enhanced ChainMail algorithm has been proposed.

4 System Overview

An overview of our deformation system is depicted in figure 2. Initial deformation input is provided by user interaction through a pick and drag interface (see section 7). The user input is processed by an extended ChainMail solver (see section 5.1) which computes a preliminary but fast deformation. The result can be visualized immediately but it is also forwarded to the relaxation solver which is initialized with the deformed voxels. Our relaxation method (section 5.2) optimizes the deformation for more realistic material behavior, but since relaxation is a time consuming iterative process, this routine is invoked in a second thread on the CPU and parallel to the rendering step performed on the GPU. Rendering is done by GPU-based direct volume raycasting (section 6). To do so, the deformed volume data needs to be resampled into a rectilinear grid and has to be transferred into graphics card memory. Since resampling is a time consuming task as well, and the amount of data that has to be downloaded to graphics memory should be as small as possible, only the changed area of the volume will be considered. For this reason both deformation methods provide bounding boxes which describe the affected part of the volume.

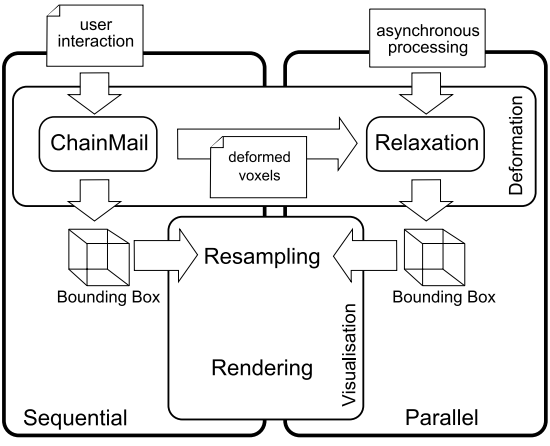


Fig. 2. Workflow

The deformation and rendering cycle performs as follows. At the begin of each loop it is checked if the user is actively manipulating the volume. In this case ChainMail deformation and visualization is performed sequentially. In the other case the relaxation solver is invoked in parallel to the rendering routine.

5 Two-Step Deformation

The proposed deformation system allows processing of large volume data sets with inhomogeneous materials. The system has been realized as two-step deformation system providing in the first step a rough and fast ChainMail based deformation, followed in a second step by a successive refinement based on a physically motivated relaxation scheme.

5.1 Step 1: Enhanced ChainMail

The first step of our deformation system is based on the *Enhanced ChainMail* [16] algorithm that allows handling of inhomogeneous data, i.e. the definition of different deformation properties per material (see section 5.3).

The ChainMail deformation process performs in the same way as proposed in the original literature, but extensions have been developed concerning data structure and data handling.

Data Structure and Memory Management. Basically we use a similar data structure as presented by Gibson et al. The original volume data (in most cases two bytes per voxel) is wrapped with an explicit position, a unique id, neighborhood information, a time stamp and flags. In our implementation we came up with a data structure using 64 bytes for one voxel.

In contrast to the original implementations we consider much larger datasets (up to 1GB), hence preparing the whole volume dataset for deformation can easily reach the limit of available main memory. Therefore we extended the data structure with a bricking scheme to reduce the allocation in main memory, similar to the method for GPU based volume rendering provided by [24]. The volume is subdivided in small bricks, $32 \times 32 \times 32$ in size. These data blocks are generated only if they are needed for the deformation process.

The data structure is controlled by a memory management algorithm. This algorithm keeps track of the available and already allocated memory. Every time data for deformation is missing the memory manager is invoked to generate the block which contains the missing data.

If the system runs out of memory an unused data block is freed before generating the new one. In this way the available memory does not limit the possible volume size but the number of data blocks which can be deformed at one time.

5.2 Step 2: Relaxation

As described in section 3, ChainMail comes with the advantage of simplicity and speed but generates only a very coarse approximation of soft body deformations. The relaxation step described in this section is suited to handle inhomogeneous data, and improves the previous deformation result of the Enhanced ChainMail algorithm by adding additional physical constraints.

Our goal was to implement a relaxation system providing physically plausible improvements of the initial ChainMail deformations while still performing at interactive frame rates. For this purpose we prpose a relaxation scheme similar to the approach presented in [2] that is directly derived from physically-based mass-spring methods. Our system based on the following basic relaxation step:

$$\mathbf{F}_i^{t+\Delta t} = D(\mathbf{p}_i^t) \tag{1}$$

$$\mathbf{p}_i^{t+\Delta t} = \mathbf{p}_i^t + \alpha \cdot \mathbf{F}_i^{t+\Delta t} \tag{2}$$

A displacement function D is evaluated to calculate a vector $\mathbf{F}_i$ which moves the node into the equilibrium position. Material properties are modeled with this displacement function. Then the node position $\mathbf{p}_i$ is updated with $\mathbf{F}_i$ scaled by a step size $\alpha < 1$. High values for α lead to faster convergence but can lead to instability as well.

Displacement Function D. The behavior of material is modeled using linear springs to connect the elements. Linear springs are described by *Hooke's Law* as

$$\mathbf{f}_{i,j}^{t+\Delta t} = -k_{i,j} \cdot (\mathbf{p}_i^t - \mathbf{p}_j^t) \tag{3}$$

where i denotes the node and j the neighbor. The constant $k_{i,j}$ denotes the stiffness of the spring between node i and j. Since the spring tries to preserve a defined distance between the nodes, the rest length d_{ij} is introduced into the equation.

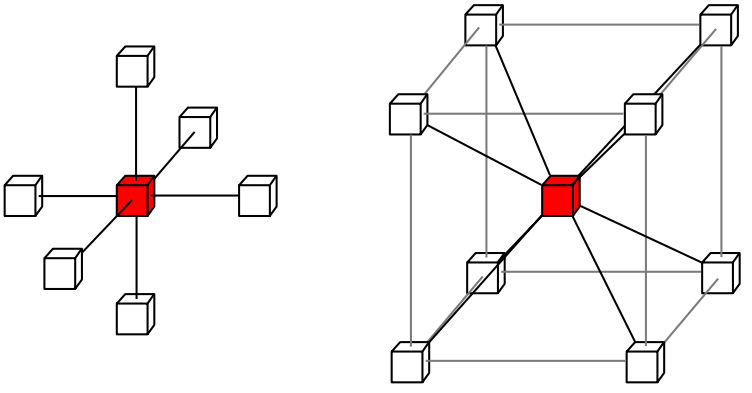


Fig. 3. Spring placement, axis aligned (structural) springs left and diagonal springs right

$$\mathbf{f}_{i,j}^{t+\Delta t} = -k_{i,j} \cdot (d_{i,j} - |\mathbf{p}_i^t - \mathbf{p}_j^t|) \cdot (\mathbf{p}_i^t - \mathbf{p}_j^t) \tag{4}$$

The force vector of node i is calculated by summing all springs.

$$\mathbf{F}_i^{t+\Delta t} = D(\mathbf{p}_i^t) = \sum_{j \in \sigma(i)} \mathbf{f}_{i,j}^{t+\Delta t} \tag{5}$$

The springs are spanned to the axis aligned neighbors to preserve the grid structure and to the diagonally opposite elements (see fig. 3) which introduces shear resistance and volume preservation behavior.

Node Processing. A generic relaxation algorithm iterates over all elements and solves the local constraints. In the special case of a 3D grid deformation problem we can take advantage of the fact that during one iteration an element can only influence its direct neighbors.

The relaxation process performs on a list of elements which are affected by the deformation. The list is initialized with all elements the previous ChainMail step has touched. During relaxation elements can be deleted from the list if a certain convergence criterion is fulfilled. In our implementation we use the length of the movement that has to be below some threshold: $\left|\mathbf{p}_i^t - \mathbf{p}_i^{t-\Delta t}\right| < \epsilon$. If convergence for the given element is not reached, all direct neighbors are added to the list. To avoid adding nodes more than once, a special flag in the element data structure shows if the node is in the list or not. A single linked list is used because this implementation has the smallest overhead for inserting and deleting items.

The advantage of processing the nodes in a wave propagation order during relaxation is discussed in [2]. Our implementation, uses this advantage without any overhead: The relaxation algorithm is initialized with elements collected from the ChainMail routine. Since ChainMail also works in a wave propagation order no additional sorting has to be done.

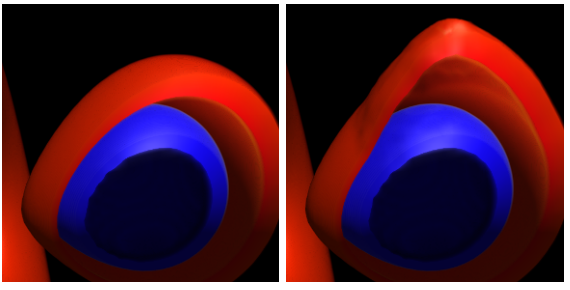


Fig. 4. Volume deformation with inhomogeneous material. The inner sphere remains undeformed because of the very soft (invisible) padding to the outer sphere.

5.3 Material Definition

The material specifications with ChainMail and relaxer properties are managed in a global list. The assignment is done through a lookup table which takes the voxel value as key.

Using a lookup table has the advantage that material properties can be easily modified while the system is running.

Applying material specifications via voxel values has similarities to transfer functions for volume rendering. In this case ranges of key values share one material. Interpolation of the parameters is not yet implemented but considered as future work.

The properties of each material have to be defined for the ChainMail solver and for the relaxation solver. Both deformation settings should produce as similar results as possible since the relaxer converges faster if ChainMail provides a good start configuration. A simple mapping from ChainMail constraints to spring properties can be designed by directly relating the size of a valid region with the stiffness of related springs.

6 Visualization

As a result of deformation the volume data is organized in an unstructured grid. Different methods for rendering scattered data have been developed which can be categorized in direct and indirect methods. *Direct* rendering of a deformed volume dataset can be done using the *projected tetrahedra* algorithm [19,22], the point-based approach called *splatting* [21,13] or a texture-based approach presented in [15,23].

In contrast, *indirect* methods require a resampling step to transform the unstructured data into a rectilinear volume representation which can be rendered using any direct volume rendering technique.

In this work we have chosen the indirect method because GPU-accelerated direct volume rendering is outclassing other volume rendering methods in quality and performance.

6.1 Resampling

Weiler et al. presented in [20] an efficient *object-order* resampling approach. This method is based on simple rasterization of a volume that is defined by tetrahedra. The subdivision of the deformed grid in tetrahedra would result in five times more (and smaller) tetrahedrons than voxels. This induces that iterating the destination voxels (in *image order*) is more efficient than iterating over tetrahedra.

Therefore we developed a new *image-order* resampling algorithm. This makes it necessary to solve the *point location problem* emphasized in [20] , i.e. to find the influencing elements in the deformed dataset corresponding to a discrete position in destination the domain.

For each voxel in the destination volume two steps have to be performed. First, find the (deformed) element that is placed next to the resampling position (the nearest neighbor). Second, compute an interpolation for the resampling position.

Nearest Neighbor Search. The first step is solved by an incremental search algorithm which traverses the deformed grid toward the resampling position. The algorithm starts with an initial element and tries to find in each iteration step an element in the local neighborhood that is placed nearer to the goal. For this algorithm it is important that the grid remains in a status where this *steepest descent*-like optimization method does not get stuck in a local minimum.

Local minima arise if the grid structure is internally overlapping as a result of deformation. Theoretically, both deformation methods keep local neighborhood relationships and prevent the grid from overlapping. Especially the ChainMail algorithm defines very strict constraints which limit the relative position of the neighbor elements. Practically, ChainMail is optimized for speed and not for accuracy and overlapping might occur in some cases. Hence the deformation system can easily produce grid configurations where the *naive* implementation fails to find better elements in the neighborhood. To escape from this "local minima" we introduce two heuristics:

- Start the search algorithm with an *estimated jump*, i.e. performing a number of traversing steps over the most promising neighbor links if the current node can not be the nearest neighbor because of its distance to the goal.
- If no better element can be found, check if it is possible for the element to be the nearest neighbor. If not, perform an *estimated jump*.

At the beginning of the resampling step an initial start point for the search is needed. We are using simply the element which would be the nearest neighbor in the undeformed grid. The resampling algorithm is performed line wise, hence local coherence can be exploited by using the last nearest neighbor as start point for the next search.

The presented search algorithm finds the nearest neighbor in more than 99% of all cases. Rarely, small resampling artifacts can be observed because of a failed search but in exchange the search algorithm performs with a almost constant complexity if local coherence is exploited.

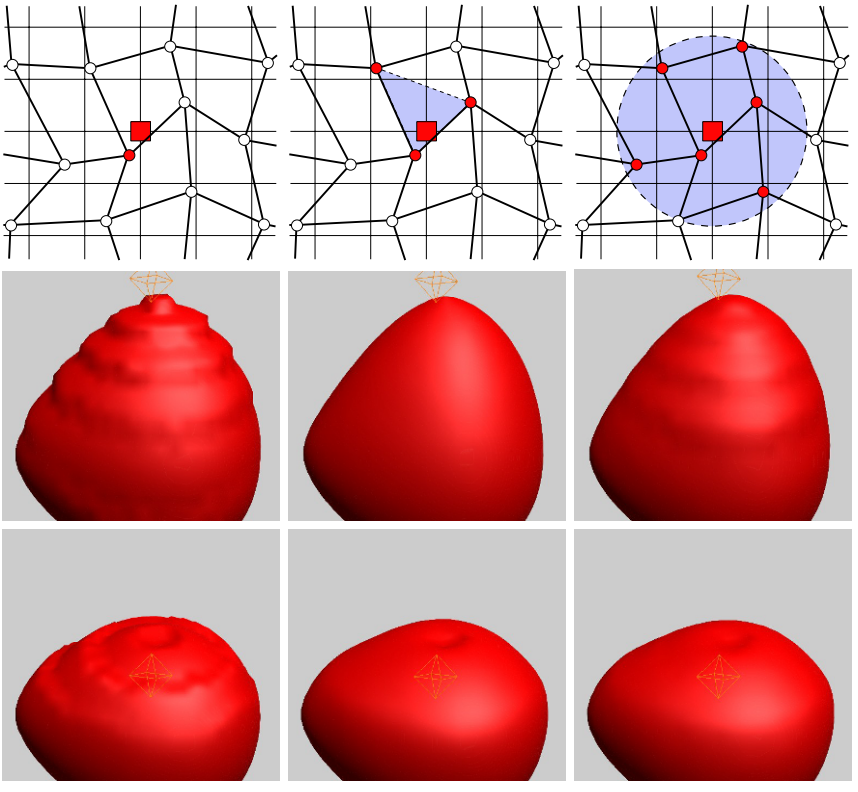


Fig. 5. Comparison: First column nearest neighbor, second column barycentric interpolation, third column radial basis functions with a radius of 1.4 and gaussian weight distribution. The first row shows a 2D simplification of the used interpolation method.

Interpolation. Once the nearest neighbor is found the value for the resampled voxel is computed. In addition to a method that directly uses the nearest neighbor value (nearest neighbor resampling) two interpolation methods have been implemented. Figure 5 shows rendering results after deformation and resampling. In the second row the visible material is expanded which means the space between the nodes is bigger as usual. In the third row we see a compressed volume where the nodes stick more together.

Optimization. To save computation time only deformed parts of the volume are resampled: While deforming the smallest bounding box enclosing all moved voxels is tracked, and transferred to the GPU for visualization.

6.2 Rendering

The actual rendering is done through GPU accelerated direct volume ray casting (refer to [17] for further details). The graphics card memory is initialized with the original volume data. Both steps of the deformation (ChainMail and relaxation) send their results to the GPU and successively replace the original volume by the resampled parts

of the deformed volume for visualization. The update is done after each iteration step. In this way only small parts of the volume have to be replaced and the exchange of the volume has no influence on the rendering speed which stays interactive during the whole deformation process.

7 User Interaction

For user interaction a simple mouse pick and drag interface was implemented. Picking is done through first hit raycasting. The mouse click position in window space is transformed into a 3D ray. This ray is traced through the volume. If a voxel value is found bigger than some given iso value, the algorithm stops and a hit point is found.

It has been shown to be quite difficult to find a proper iso value for picking if volume rendering with complex transfer functions is used. Therefore a special rendering mode which combines direct volume rendering with first hit raycasting [17] allows visualization and adaptation of a chosen iso-surface on the fly For picking, the same iso value is used and the hitpoint corresponds to the visual feedback.

8 Results

For timing tests an intel dual core machine with 2.4 GHz, 2 GB of RAM and a Nvidia Quadro FX 3400 graphics adapter with 256 MB RAM was used. Table 1 shows the overall performance of the system stated in framerates. The listed results prove that the deformation system remains interactive even if very large datasets are used. The results have been produced in normal use case situations with up to 1 million voxel per deformation such as the examples shown in figure 6.

The execution speed has been measured for the sub-modules of the deformation system as well. The ChainMail algorithm reaches an average performance of $2.34 \cdot 10^6 \frac{elements}{second}$, the computation cost is growing linearly with the number of deformed elements.

The performance of the resampling algorithm depends on the used interpolation method. Nearest neighbor resampling reaches an average performance of

Table 1. Timing tests

data set	*dimension*	*size*	*rendering only*	*rendering + ChainMail*	*rendering + relaxation*
hydrogen	64×64×64	0.5 MB	31.1 fps	30 fps	15.50 fps
endoscopy	512×512×128	64 MB	11.3 fps	9.5 fps	7.3 fps
head	512×512×333	166.5 MB	15.3 fps	9.1 fps	6.90 fps
beetle	832×832×494	652.1 MB	4.23 fps	3.65 fps	3.45 fps

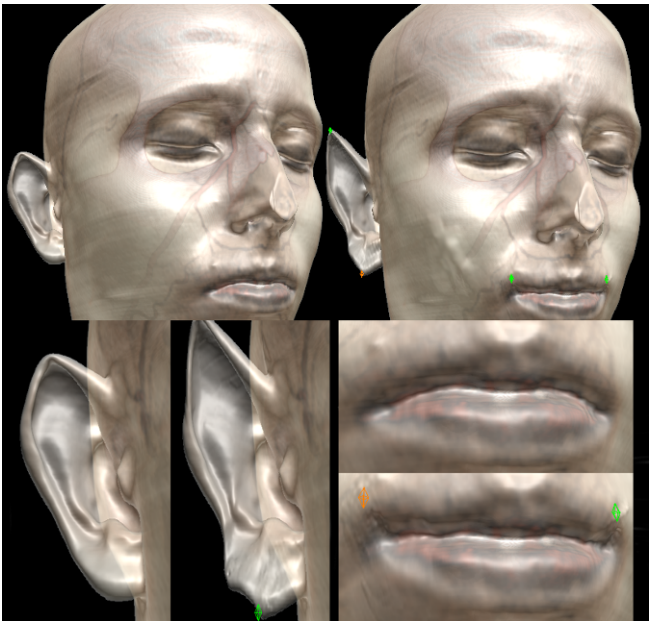


Fig. 6. Deformation of the "head" dataset containing 87.3 million voxels. Local deformations involving 2.5 million voxels (ear) and 0.7 million voxels (mouth).

$4.41 \cdot 10^6 \frac{voxel}{second}$, barycentric coordinates interpolation $2.46 \cdot 10^6 \frac{voxel}{second}$ and interpolation using RBF $1.48 \cdot 10^6 \frac{voxel}{second}$.

The performance of the relaxation step is difficult to measure. The algorithm iterates $2.3 \cdot 10^7 \frac{elements}{second}$ but each element has to be processed many times until convergence is reached. The number of needed iterations depends on the size of the deformation and on the material parameters. Tests have shown that a deformation with two million elements involved is relaxed within 3 seconds, while the system remains fully interactive.

9 Summary and Discussion

We have presented a complete system for interactive deformation of inhomogeneous volume data combined with high quality rendering. Unlike other deformation systems we perform all computations directly at the voxel level without any simplification or preprocessing. Our system has proven to be able to handle datasets with more than 650 MB (340 million voxels) while more than 1 million voxels can be interactively deformed simultaneously.

Due to the high amounts of elements that have to be deformed, the Enhanced Chain-Mail plus relaxation approach for the deformation system turned out as the only possible solution. However, the choice was a trade off between accuracy and interactivity and is not able to reach the physical exactness of finite element or mass-spring systems. The integration of more exact deformation systems is considered as future work and will

become more and more possible with increasing computational power, like the recently introduced PPUs (Physics Processing Units).

The resampling based visualization system has proven to be well suited for the given problem. The main advantage is the possibility to integrate the deformation system seamlessly into the high quality direct volume rendering framework. However, the implementation of direct approaches able to render unstructured data directly are also considered as future work to overcome the resampling overhead.

References

1. Adams, B., Keiser, R., Pauly, M., Guibas, L.J., Gross, M., Dutré, P.: Efficient raytracing of deforming point-sampled surfaces. Computer Graphics Forum 24(3), 677–684 (2005)
2. Brown, J., Sorkin, S., Bruyns, C., Latombe, J.-C., Montgomery, K., Stephanides, M.: Algorithmic tools for real-time microsurgery simulation. Medical Image Analysis 6, 289–300 (2002)
3. Chen, M., Correa, C., Islam, S., Jones, M.W., Shen, P.-Y., Silver, D., Walton, S.J., Willis, P.J.: Deforming and animating discretely sampled object representations. In: Chrysanthou, Y., Magnor, M. (eds.) Eurographics - State of the Art Reports, pp. 113–140 (2005)
4. Chen, M., Silvery, D., Winter, A.S., Singhy, V., Cornea, N.: Spatial transfer functions - a unified approach to specifying deformation in volume modeling and animation. In: Fujishiro, I., Mueller, K., Kaufman, A. (eds.) Proceedings of Volume Graphics, pp. 35–44. The Eurographics Association (2003)
5. Gibson, S., Fyock, C., Grimson, E., Kanade, T., Kikinis, R., Lauer, H., McKenzie, N., Mor, A., Nakajima, S., Ohkami, H., Osborne, R., Samosky, J., Sawada, A.: Simulating surgery using volumetric object representations, real-time volumerendering, and haptic feedback. Medical Image Analysis 2(2), 121–132 (1998)
6. Gibson, S.: Beyond volume rendering: Visualization, haptic exploration, an physicalmodeling of voxel-based objects. In: Scanteni, R., van Wijk, J., Zanarini, P. (eds.) Proceedings of Visualization in Scientific Computing, pp. 9–24. Springer, Heidelberg (1995)
7. Gibson, S.: 3D chainmail: A fast algorithm for deforming volumetric objects. In: Proceedings of Symposium on Interactive 3D Graphics, pp. 149–154 (1997)
8. Gibson, S.: Using linked volumes to model object collisions, deformation, cutting, carving,and joining. IEEE Transactions on Visualization and Computer Graphics 5(4), 333–348 (1999)
9. Georgii, J., Westermann, R.: A multigrid framework for real-time simulation of deformable bodies. Computers and Graphics 30(3), 408–415 (2006)
10. Li, Y., Brodlie, K.: Soft object modelling with generalised chainmail - extending the boundaries of web-based graphics. Comput. Graph. Forum 22(4), 717–728 (2003)
11. Mosegaard, J., Herborg, P., Sorensen, T.S.: A gpu accelerated spring mass system for surgical simulation. In: Westwood, J.D., Haluck, R.S., Hoffman, H.M., Mogel, G.T., Phillips, R., Robb, R.A., Vosburgh, K.G. (eds.) 13th Medicine Meets Virtual Reality Conference, volume 111 of Studies in Health Technology and Informatics, pp. 342–348. IOS Press (2005)
12. Müller, M., Heidelberger, B., Teschner, M., Gross, M.: Meshless deformations based on shape matching. ACM Transactions on Graphics 24(3), 471–478 (2005)
13. Neophytou, N., Mueller, K.: Gpu accelerated image aligned splatting. In: Proceedings of Volume Graphics, pp. 197–205 (2005)
14. Nealen, A., Müller, M., Keiser, R., Boxerman, E., Carlson, M.: Physically based deformable models in computer graphics. Eurographics - State of the Art Reports, 71–94 (2005)

15. Rezk-Salama, C., Scheuering, M., Soza, G., Greiner, G.: Fast volumetric deformation on general purpose hardware. In: Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics Hardware, pp. 17–24. ACM Press, New York (2001)
16. Schill, M., Gibson, S., Bender, H.-J., Männer, R.: Biomechanical simulation of the vitreous humor in the eye using an enhanced ChainMail algorithm. In: Wells, W.M., Colchester, A.C.F., Delp, S.L. (eds.) MICCAI 1998. LNCS, vol. 1496, pp. 679–687. Springer, Heidelberg (1998)
17. Scharsach, H., Hadwiger, M., Neubauer, A., Wolfsberger, S., Büuhler, K.: Perspective isosurface and direct volume rendering for virtual endoscopyapplications. In: Proceedings of Eurovis 2006, pp. 315–322 (2006)
18. Singh, V., Silver, D., Cornea, N.: Real-Time Volume Manipulation. In: Proceedings of the Eurographics/IEEE TVCG Workshop on Volume Graphics, vol. 45, pp. 45–51 (2003)
19. Shirley, P., Tuchmany, A.: A Polygonal Approximation to Direct Scalar Volume Rendering. In: Proceedings of Workshop on Volume Visualization, vol. 24(5), pp. 63–70 (1991)
20. Weiler, M., Ertl, T.: Hardware-software-balanced resampling for the interactive visualization of unstructured grids. In: Ertl, T., Joy, K., Varshney, A. (eds.) Proceedings of IEEE Visualization, pp. 199–206 (2001)
21. Westover, L.: Footprint evaluation for volume rendering. In: Proceedings of ACM SIGGRAPH, pp. 367–376 (1990)
22. Weiler, M., Kraus, M., Merz, M., Ertl, T.: Hardware-based view-independent cell projection. IEEE Transaction on visualization and Computer Graphics 9(2), 163–175 (2003)
23. Westermann, R., Rezk-Salama, C.: Real-time volume deformations. Computer Graphics Forum 20(3), 443–451 (2001)
24. Weiler, M., Westermann, R., Hansen, C.D., Zimmerman, K., Ertl, T.: Level-of-detail volume rendering via 3D textures. In: IEEE Symposium on Volume Visualization and Graphics, pp. 7–13 (2000)

Part III
Interactive Environments

A Multi-resolution Mesh Representation for Deformable Objects in Collaborative Virtual Environments

Selcuk Sumengen, Mustafa Tolga Eren, Serhat Yesilyurt, and Selim Balcisoy

Sabanci University, Faculty of Engineering and Natural Sciences TR-34956 Orhanli
Tuzla - Istanbul, Turkey
{selcuk,mustafae}@su.sabanciuniv.edu,
{syesilyurt,balcisoy}@sabanciuniv.edu
http://graphics.sabanciuniv.edu

Abstract. This paper presents a method for physical simulation of deformable closed surfaces over a network, which is suitable for realistic interactions between users and objects in a Collaborative Virtual Environment (CVE). To demonstrate a deformable object in a CVE, we employ a real-time physical simulation of a uniform-tension-membrane, based on linear finite-element-discretization of the surface. The proposed method introduces an architecture that distributes the computational load of physical simulation between each participant. Our approach requires a uniform-mesh representation of the simulated structure; therefore we designed and implemented a re-meshing algorithm that converts irregularly triangulated genus zero surfaces into a uniform triangular mesh with regular connectivity. The strength of our approach comes from the subdivision methodology that enables to use multi-resolution surfaces for graphical representation, physical simulation, and network transmission, without compromising simulation accuracy and visual quality.

Keywords: Deformable objects, real-time simulation, cloth modelling, Distributed and Network Virtual Environments, Collaborative Virtual Environments.

1 Introduction

Collaborative Virtual Environments (CVE)s are being extensively used for training, design and gaming for several years. They enable participants to get immersed into a Virtual Environment where they can perform a task or experience a story together. In most use cases such as gaming and education, current CVEs are sufficient to address user expectations related to visual realism, animations and networking. However, CVEs also involve substantial amount of interaction between the users and the objects in synthetic worlds, which should be visually appealing and physically realistic as well. Current CVEs are mostly limited to avatar-avatar interaction or the object interactions are animated using offline techniques and they are commonly hard-coded into the application. Another recent approach is to use rigid body simulations together with inverse kinematics engines [1]. Real-time physical simulation of deformable bodies in CVEs will enable accurate replication of interaction with real world deformable objects and open

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 75–87, 2008.

a vast array of possible applications. One example is medical and engineering applications which require accurate simulations in real-time.

In this paper, we are presenting a method for deformations on closed surfaces over a peer-to-peer network architecture (Fig. 1).

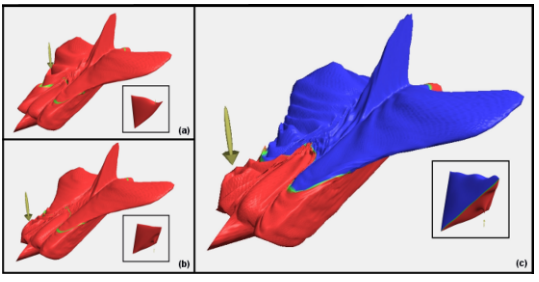


Fig. 1. First (a) and second (b) peers connected to the Virtual Environment, deforming a sample deformable model. (c) Colors red and blue denote domains of different peers in a collaborative deformation.

2 Related Work

2.1 Collaborative and Distributed Network Virtual Environments

DIVE [2] is one of the first Distributed Virtual Environments that allows participants to collaborate in a 3D virtual world which facilitates audio, video and text transmission for communication and interaction within the VE. Similarly, NPSNET [3] is designed for military training and simulation for networked environments using Distributed Interactive Simulation Standard (DIS). MASSIVE is a VR conferencing system especially used for public participation and performance [4]. VLNET allows multiple users represented by 3D virtual human actors to interact with each other and enables third parties to view the shared virtual environment from the Web using VRML[5].

There are only a few systems that in particularly deal with the significance of physical simulation in collaborative virtual environments. A recent work by Jorissen [1], gives a detailed survey on state of the art of dynamic interactions and physical simulations in CVEs. Jorissen et al. introduces a collaborative virtual environment, where the object-object interaction is allowed in addition to avatar-object and avatar-avatar interactions using a non-commercial physics engine.

There are few attempts to introduce deformable objects into CVEs: Dequidt et al. [6] propose a system based on ghost objects to handle network latency. Ghost objects are associated to objects manipulated over the network and introduced into the client side to perform physical simulations asynchronously at each user.

Collaborative Haptics Environments are also introduced to handle surgical training and simulations [7]. As haptic rendering must be performed at simulation rates higher than 1 KHz, most systems require dedicated hardware running on real-time operating systems [8]. Goncharenko et al. [9] report a distributed and collaborative haptic visualization of

a 1-DOF crank model only possible on Intranets. They used a dedicated haptic communication library to satisfy real-time communication requirements of haptic rendering on a client-server architecture connected through Ethernet.

2.2 Deformable Objects

Visualization of object deformations is an important research area for over two decades with a large span of applications such as cloth, tissue modeling and virtual surgery. One set of approaches on the visualization of deformable models is non-physical and purely geometric techniques, most of which is classified as Free-Form-Deformations [10]. Physics based approaches gained a popular attention by enabling cloth animations [11]. Cloth animation is an extensive research area covering wide range of issues from physical simulation to collusion detection [12]. Early examples of cloth animation using a linear model based on energy minimization, and continuing approaches using explicit integration schemes, are suffering from stability issues for large body deformations. Baraff and Witkin [13], introduced an implicit integration scheme for stable simulations using large time steps. On the other hand, real-time simulation of deformable models is an other challenge, and linear mass-spring models introduced at first [14]. As an alternative, Boundary Element Method is introduced, which is inspired by Finite Element Method (FEM), however, considers only the surface of the model [15]. Non-linear FEMs are not suitable for real-time simulations since they are computationally intensive, so deformable object simulations in virtual environments continued to use improved massspring models [16]. Also, precomputed models for real-time dynamic deformations are considered [17]. Since medical applications require real-time and accurate simulations some approaches used FEM to parameterize the mass-spring model to improve accuracy [18].

3 Network Deformable Objects

Our method applies a collaborative deformation on a linear membrane model over network, which can be used for simulation of deformable objects (tissue, organ, cloth) in CVEs.

3.1 Geometric Model

The proposed approach requires a uniform representation of the simulated structure. Restriction on the genus of the model allows us to construct a regular 2D grid that corresponds to the surface of the model.

The surface of any convex polyhedron is homeomorphic to a sphere and has Euler characteristic of two. Homeomorphic spaces are identical from the viewpoint of the topology therefore genus zero surfaces preserve their topological properties under spherical parameterization and can be mapped onto a convex regular polyhedron.

Mesh Representation. We have chosen Tetrahedron as the Domain for our mesh representation, since it has four equilateral triangular faces that can be represented as a 2D

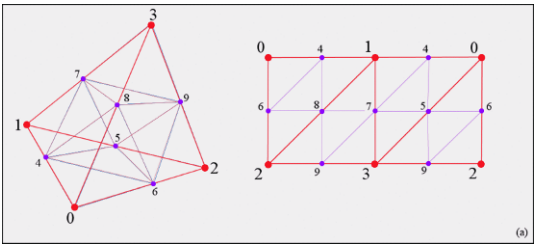


Fig. 2. 2D Grid representation of a tetrahedron

grid having $(2^n + 1)x(2^{n+1} + 1)$ nodes where, n is positive integer determining the number of vertices and will be referred as detail level (Fig. 2).

Mesh Generation. We propose an algorithm that converts irregularly triangulated genus zero surfaces into a uniform mesh with regular connectivity. Previous approach for constructing regular meshes with fixed and simple topology by Hoppe [19], generates a spherical parameterization of the surface and the domain. Surface, projected on the sphere, mapped on to the domain, and unfolded to generate the geometry image. We apply a similar procedure, but we introduce a different technique for spherical parameterization and model re-meshing. It allows adjusting the tradeoff between face area uniformity of the generated mesh, and preserving the accuracy with the original mesh.

Given a triangle mesh M, the problem of spherical parameterization is to form a continuous invertible map $\psi : S \rightarrow M$ from the unit sphere to the mesh [19]. Spherical parameterization of both a regular tetrahedral domain D and an irregular input mesh M are necessary to generate Sphere to Mesh $(S \rightarrow M)$ and Sphere to Domain $(S \rightarrow D)$ mappings that will allow us to perform Mesh to Sphere and Sphere to Domain $(M \rightarrow S \rightarrow D)$ transformation.

Any convex polyhedron can easily be projected onto a unit sphere (Fig. 3) by switching to spherical coordinate system (θ, ϕ, r) and setting a unit radius for all vertices (Gnomonic Projection), however translation between each mesh triangle and spherical triangle might introduce a certain amount of distortion.

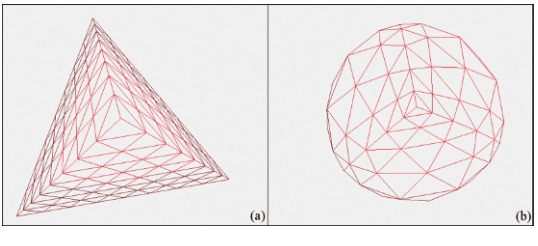


Fig. 3. Gnomonic projection of a tetrahedron

Previous approaches define a stretch norm to measure the stretch efficiency and conclude that minimizing the stretch norm is a non-linear optimization problem [19,20]. We attack this problem by a modification of a well known technique used for graph drawing. Graph drawing using force directed placement methods, which are also called springembedders, distributes vertices evenly in the frame and minimize edge crossings while favoring uniformity of the edge lengths [21]. Since we implemented a deformable physics engine that can handle mass spring systems efficiently, we introduce a variant of spring-embedders for stretch optimization.

A spring-embedder model is generated from the gnomonic projection of the domain. Every vertex has a constant mass, and springs are introduced between neighboring vertices. An external force field (Eq. 1) is applied from the center of the domain that limits displacements of vertices on the unit sphere.

$$f_{external_i} = (1 - ||x_i||) \times \hat{x}_i \qquad \forall i\,,\ 0 \leq i \leq nNodes\,. \tag{1}$$

Springs between the vertices tend to preserve initial edge lengths and resist movements that change the topology; however we need to establish a tension on these springs to perform stretch optimization.

We scale down the positions of the vertices that are projected onto unit sphere (Eq. 2), and an external force which is applied continuously expands the vertices onto the unit sphere again while producing a tension on the springs. Stiffness parameters are updated continuously to achieve an area uniform tessellation over the unit sphere (Fig. 4).

$$x_{i_{new}} = C \times x_i \qquad \forall i\,,\ 0 \leq i \leq nNodes\,,\ 0 \leq C \leq 1\,. \tag{2}$$

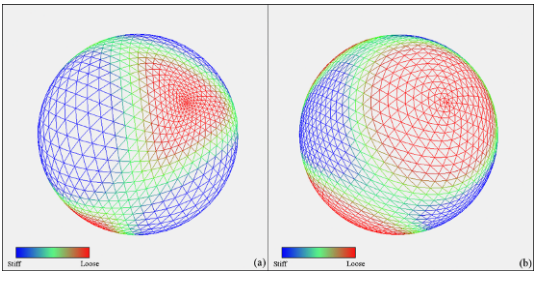


Fig. 4. (a) Gnomonic Projection of Tetrahedron. (b) Stretched Gnomonic Projection of Tetrahedron.

Our proposed force model is a feasible stretch optimization technique for domain to sphere mapping; however, it is insufficient for mesh to sphere mappings where the projection of nonconvex polyhedron into a unit sphere results in edge crossings and does not preserve initial surface topology. We use a vertex displacement procedure (Eq. 3) which is similar to the relaxation method of previous spherical parameterization approaches [22] to overcome this problem (Fig. 5).

$$x_{i_{new}} = \sum_{j=0}^{nNeighbors_i} \frac{x_{ij}}{nNeighbors_i}\,, \quad \forall i\,,\ 0 \leq i \leq nNodes\,, \quad x_{ij}\ is\ j_{th}\ neighbor\ of\ x_i\,. \tag{3}$$

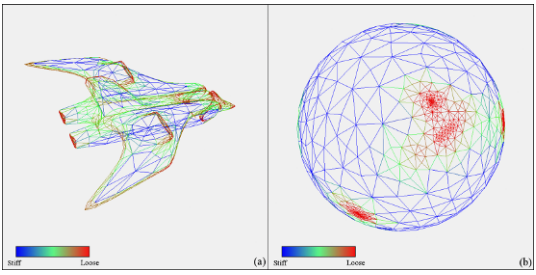


Fig. 5. (a) Irregular Input Mesh. (b) Stretched Gnomonic Projection of Input Mesh.

Model Re-meshing. Combining the spherical mappings mesh to sphere ($M \rightarrow S$) and sphere to domain ($S \rightarrow D$) to derive mesh to domain mapping ($M \rightarrow D$), requires intersection of the sets on the sphere. However, transformed vertex coordinates of the mesh and domain might not intersect on the sphere, and vertices of the domain might fall inside of a mesh facet. For each vertex of the domain, intersecting face of the parameterized mesh should be found out and 3D coordinates of domain vertex should be computed by interpolating the vertices of the intersecting face (Fig. 6).

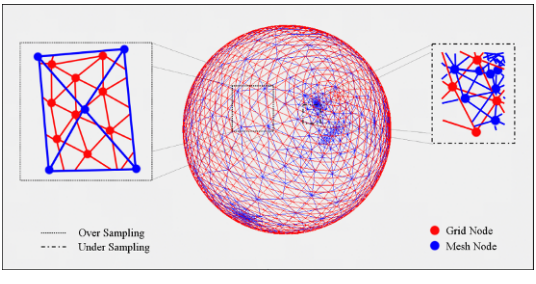


Fig. 6. Intersecting Spherical Projections of Tetrahedral Domain and Input Mesh

Since computing the interpolated coordinates is costly, we introduce a fast method taking advantage of recent advances in graphics hardware using the GPU and frame buffer objects. Using OpenGL and programmable shaders (GLSL), we render the faces of the parameterized mesh onto the frame buffer using the two dimensional spherical coordinates (θ and ϕ) of the transformed vertices. Initial Cartesian coordinates (x, y, and z) of the parameterized mesh vertices are attached to color attributes (r, g, and b) at the vertex shader, and inside of each face is filled with the interpolated Cartesian coordinates at the fragment level (Fig. 7). Rendered image is then fetched from the frame buffer as a 2D texture and used like a lookup table to generate 3D coordinates of the domain vertices(Fig. 8).

Subdivision Scheme using Convolution Kernels. Subdivision methodology is appropriate for our approach since it allows multi-resolution representation of a surface and

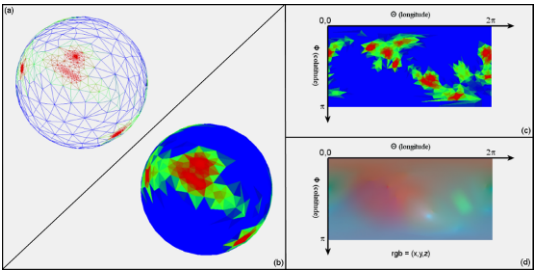


Fig. 7. Spherical projection of input mesh is, (a) rendered as 3D wireframe, (b) 3D colored surface, (c) 2D colored surface, and (d) 2D colored surface, where the original positions of vertices are used as color components

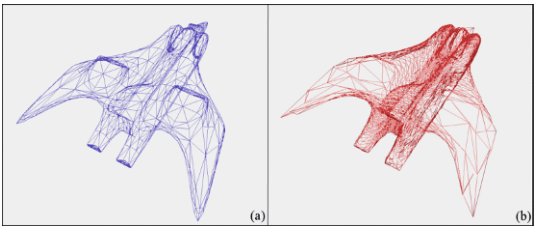


Fig. 8. Final comparison of (a) the input mesh with 1444 vertices, and (b) the resulting regular mesh with 8385 vertices

fast switching between detail levels. It also favors numerical stability, so it is highly suitable for physical simulation of deformations using finite element and finite difference methods.

We used a variant of butterfly subdivision scheme [23] that generates a C^1 smooth triangular mesh. Modified Butterfly Scheme is an interpolating subdivision scheme, where the original vertices (control points) are also the vertices of the refined surface and surface is interpolating to a limit surface. This behavior makes it possible to use surfaces with different resolutions for graphical representation, physical simulation, and network transmission, without compromising the integrity of simulation accuracy and the rendered image.

Given that we have a regular mesh representation as a grid structure, we introduce some modifications (Fig. 9) to apply a fast and robust refinement strategy using modified butterfly scheme. Taking advantage of having a regular domain, we have no boundary or crease vertices, but there are four extraordinary vertices of valances three on the corners of the tetrahedral domain. However, if we duplicate the edges of these vertices, they can be treated as regular vertices. Since the duplicate edges are symmetric to existing edges, resulting odd vertices will have same values. This modification allows us to use the mask for interior odd vertices with regular neighbors for all the grid nodes. We also introduce offsets to 2D grid representation. Offsets are the copies of grid nodes, assuring existing neighboring properties and they are kept updated before the convolution process. Having a 2D grid representation and a mask with constant coefficients, odd vertices can be generated by consecutive convolutions with three kernels created

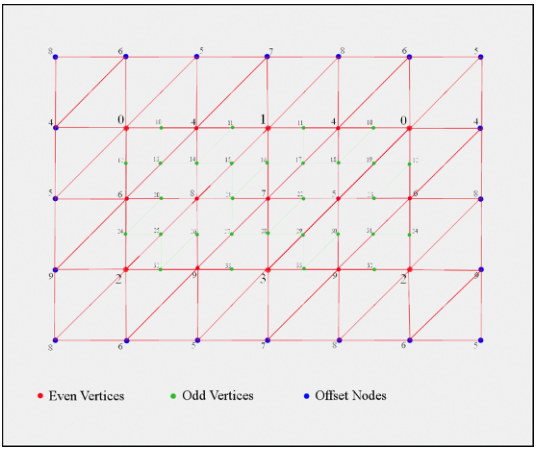


Fig. 9. Modified 2D Grid Structure

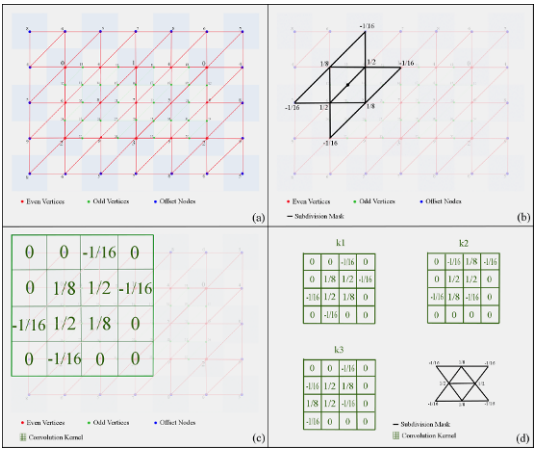


Fig. 10. (a) Modified 2D Grid Structure. (b) Application of mask for interior odd vertices with regular neighbors. (c) Equivalent convolution kernel. (d) Three convolution kernels generated for three edges.

by rotating the subdivision mask three times (Fig. 10). Necessity for the grid offsets arises from the application of the mask to the grid boundaries, and modified subdivision scheme requires first neighbors of even vertices that are next to generated vertex. Offset width does not change according to the grid dimensions and time required for the update of the offsets is negligible. After the convolution of the nth level subdivision surface three times, resulting 2D grids are merged to generate $(n+1)_{th}$ level subdivision surface having $(2^n+1) \times (2^{n+1}+1)$ nodes (Fig. 11).

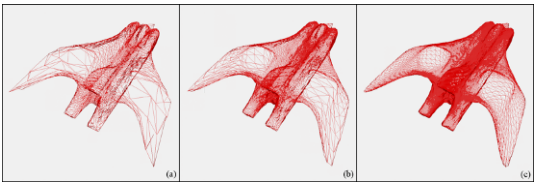


Fig. 11. Comparison of resulting mesh refined by subdivision and rendered at different level of details: (a) 8335 vertices, (b) 33153 vertices, (c) 131841 vertices

3.2 Physical Model

Physical simulation of deformable objects is an extended research area, where several methods are present, varying from fast and simple methods favoring speed and scalability, to much more complex methods favoring accuracy and stability. Linear methods such as mass-spring models for dynamic deformations are suitable for use in realtime applications; however, they are not capable of handling large deformations and small time steps which are required to guarantee stability [14,24]. On the other hand, nonlinear models incorporating large viscoelastic and plastic deformations are computationally intensive [25], and despite their physical accuracy, real-time simulation of large deformations is only possible with massively parallel computers.

For the demonstration of the deformable object on a collaborative virtual environment, we use a real-time physical simulation of a uniform-tensionmembrane, based on linear finite-elements. We introduced finite element discretization to form the global stiffness matrix, which is updated frequently to handle large deformations with enhanced accuracy and we used Runge-Kutta-Fehlberg method for integration to achieve bigger time steps and improved stability [26].

Linear Finite-Element Model. Application of the finite-element method for the wave equation [27,25], describing the time-dependent small deformations of a uniform-tension membrane results in a standard system of equations [28] (Eq. 4):

$$M\ddot{x} \;=\; -B\dot{x} \;-\; Kx \;+\; f_{external}\,. \tag{4}$$

where, x is the normal deformation of each node, M is the diagonal mass matrix, $f_{external}$ is the external force vector due to user interactions, B is the diagonal damping matrix, and K is the stiffness matrix. In our implementation, we separate normal deformation and the velocity of each node to improve the stability of the Runge-Kutta method used to solve the linear system. Namely, we have (Eq. 5),

$$\dot{x} \;=\; v\,, \tag{5}$$

and the resulting equation of motion (Eq. 6):

$$M\dot{v} \;=\; -Bv \;-\; Kx \;+\; f_{external}\,. \tag{6}$$

The finite element method works well with an arbitrary triangulation of a surface as well as proposed regular grid structure. In our implementation we apply the damping

matrix directly on the nodal velocities, so as to model a permeable membrane placed in a liquid. In some standard formulations, the damping is applied to relative nodal velocities. The two yields in similar solutions, however our implementation results in simpler sparse structures and faster simulation times via improved stability of nodal damping.

3.3 Network Model

There are several network topologies used for Distributed Virtual Environments. Our approach is implemented with a peer-to-peer architecture which is operational on local and wide area networks. User Datagram Protocol (UDP) is used for communication, since speed and bandwidth requirements are essential for a real time simulation and have a greater priority over packet integrity.

Peers can run on different computers on the network or can be started in the same application as separate threads. We dont introduce any dedicated servers, and peer nodes are functioning as both clients and servers. Every peer has a listening port and address for incoming connection requests. The peer which started to run CVE is required to act as a master for coordinating partitioning of the simulation. Partitioning occurs after sending a request by a participant which selects a face on the mesh and identifies it as the point of interest where the peer is going to introduce an external force. Participants can enter the CVE also as a viewer, where they do not interact with the model, but can observe the simulation.

3.4 Partitioning and Synchronization of Physical and Geometric Models through the Network

In our approach, partitioning the deformable object and synchronizing among peers is an important issue, since it enables collaboration in the virtual environments with distributed computational load. For an efficient communication and separation, we introduce a quad tree based data structure over 2D grid structure proposed on the previous sections. Quad-tree structure (Fig. 12) is a natural formation for the tetrahedral domain, and can be divided hierarchically. Tree nodes are transferred efficiently via network since a tree node contains a range identifier which is actually the combination of upper left and lower right node index numbers, and state information of corresponding region as a 2D array. Minimum depth level for the tree can be adjusted to keep the packaged tree node size smaller then the maximum packet size allowed by the network protocol. Domain divisions are designed upon a quad tree based structure in the figure (Fig. 12). While dividing the domain into sub-domains, equivalence of the number of shared grid nodes is an important criterion. However, keeping the domain boundaries shorter for an accurate synchronization of the physical simulation is essential, and keeping the fragmentation minimal for efficient network transmission is also important.

At the beginning of the simulation, each client starts to simulate the whole domain independently. When a connection invoked, domain is partitioned according to the points of interest where the forces are applied by the clients. Nodes at the domain boundaries are treated as boundary conditions, and the dynamical simulation of the local domain performed consequently at the each client.

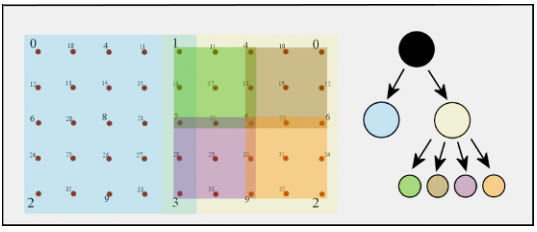


Fig. 12. Sample tree structure for tetrahedral domain having depth of two

4 Results

Our graphical sub-system can efficiently handle very large meshes, taking advantage of regular-mesh and subdivision methodology as presented in the previous chapters (Fig. 13). Our system renders meshes using the Phong shading model at interactive frame rates (25 fps) with resolution up to 100K polygons on an AMD Opteron 2.6 GHz PC equipped with NVIDIA Quadro FX4500 GPU. We implemented Phong shading model on the GPU. Vertex positions are uploaded to texture memory and vertex normals are computed on the fly using texture lookups.

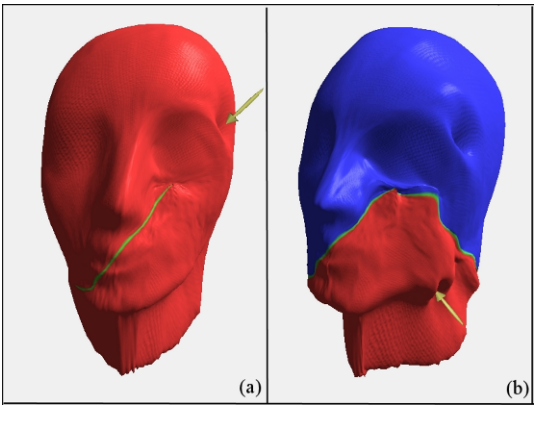


Fig. 13. (a) One peer and (b) two-peers collaborative network deformation of a sample model having a regular mesh structure

The proposed network communication model can handle synchronous simulation among two peers of a surface up to 10K vertices over the local area network. This level has a bandwidth requirement of 20 M Bits per second without any compression.

We also tested the performance of the system by comparing computational load and number of simulated nodes. Our deformation engine can handle multi-resolution meshes up to 30K nodes, and maintains interactivity at less than utilization. Partitioning the domain between clients reduces computational load by 45and increases the running speed by a factor of 1.8, depending on the partitioning ratio.

5 Conclusions

We have proposed a new technique for deformable body simulations in the field of collaborative virtual environments and introduced several improvements over the methods we adopted. We found that adaptive refinement and multilevel meshing strategies are promising research domains that can be further exploited for increased network efficiency and better physical accuracy for CVEs.

Furthermore, we showed that the partitioning of physical simulation domain has a considerable effect on performance, and makes real-time simulation possible in scenarios where only one peer is incapable of handling the computational load.

As future work, we consider on the fly compression which might significantly reduce the bandwidth requirement but can degrade overall performance because of the additional computational cost. Optimization of the system for the Internet is out of the scope of this paper, but it is safe to predict that the network lag on public networks will have an impact on performance. Our method needs to be optimized for the Internet, and tested over large physical distances to overcome possible negative network effects.

References

1. Jorissen, P., Wijnants, M., Lamotte, M.: Dynamic interactions in physically realistic collaborative virtual environments. IEEE Transactions on Visualization and Computer Graphics 11, 649–660 (2005)
2. Hagsand, O.: Interactive multiuser VEs in the DIVE system. Multimedia 3, 30–39 (1996)
3. Macedonia, M., Zyda, M., Pratt, D., Barham, P., Zeswitz, S.: NPSNET- A network software architecture for large-scale virtual environments. Presence- Teleoperators and Virtual Environments 3, 265–287 (1994)
4. Benford, S., Greenhalgh, C., Rodden, T., Pycock, J.: Collaborative virtual environments. Communications of the ACM 44, 79–85 (2001)
5. Thalmann, D., Babski, C., Capin, T., Thalmann, N., Pandzic, I.: Sharing VLNET worlds on the Web. Computer Networks and ISDN Systems 29, 1601–1610 (1997)
6. Dequidt, J., Grisoni, L., Chaillou, C.: Collaborative interactive physical simulation. In: Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia, pp. 147–150. ACM Press, New York (2005)
7. Shen, X., Bogsanyi, F., Ni, L., Georganas, N.: A heterogeneous scalable architecture for collaborative haptics environments. In: Proceedings of the 2nd IEEE Internatioal Workshop on Haptic, Audio and Visual Environments and Their Applications (HAVE 2003), pp. 113–118 (2003)
8. Zhou, J., Shen, X., Georganas, N.: Haptic tele-surgery simulation. In: Proceedings of the 3rd IEEE International Workshop on Haptic, Audio and Visual Environments and Their Applications (HAVE 2004), pp. 99–104 (2004)
9. Goncharenko, I., Svinin, M., Matsumoto, S., Masui, Y., Kanou, Y., Hosoe, S.: Cooperative control with haptic visualization in shared virtual environments. In: Proceedings of Eighth International Conference on Information Visualisation (IV 2004), pp. 533–538 (2004)
10. Sederberg, T., Parry, S.: Free-form deformation of solid geometric models. ACM SIGGRAPH Computer Graphics 20, 151–160 (1986)
11. Terzopoulos, D., Platt, J., Barr, A., Fleischer, K.: Elastically deformable models. ACM SIGGRAPH Computer Graphics 21, 205–214 (1987)

12. Volino, P., Magnenat-Thalmann, N.: Resolving surface collisions through intersection contour minimization. ACM Transactions on Graphics (TOG) 25, 1154–1159 (2006)
13. Baraff, D., Witkin, A.: Large steps in cloth simulation. In: Proceedings of the 25th annual conference on Computer graphics and interactive techniques, pp. 43–54 (1998)
14. Desbrun, M., Schroder, P., Barr, A.: Interactive animation of structured deformable objects. In: Graphics Interface 1999, vol. 1 (1999)
15. James, D., Pai, D.: ArtDefo: accurate real time deformable objects. In: Proceedings of the 26th annual conference on Computer graphics and interactive techniques, pp. 65–72. ACM Press/Addison-Wesley Publishing Co., New York (1999)
16. Kang, Y., Cho, H.: Complex deformable objects in virtual reality. In: Proceedings of the ACM symposium on Virtual reality software and technology, pp. 49–56. ACM Press, New York (2002)
17. Nikitin, I., Nikitina, L., Frolov, P., Goebbels, G., Göbel, M., Klimenko, S., Nielson, G.: Real-time simulation of elastic objects in virtual environments using finite element method and precomputed Green's functions. In: Proceedings of the workshop on Virtual environments 2002, Eurographics Association Aire-la-Ville, Switzerland, pp. 47–52 (2002)
18. Choi, K., Sun, H., Heng, P.: An efficient and scalable deformable model for virtual reality-based medical applications. Artificial Intelligence In Medicine 32, 51–69 (2004)
19. Praun, E., Hoppe, H.: Spherical parametrization and remeshing. ACM Transactions on Graphics 22, 340 (2003)
20. Sander, P., Snyder, J., Gortler, S., Hoppe, H.: Texture mapping progressive meshes. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pp. 409–416. ACM Press, New York (2001)
21. Fruchterman, T., Reingold, E.: Graph Drawing by Force-directed Placement. Software- Practice and Experience 21, 1129–1164 (1991)
22. Alexa, M.: Recent Advances in Mesh Morphing. Computer Graphics Forum 21, 173–196 (2002)
23. Zorin, D., Schröder, P., Sweldens, W.: Interpolating Subdivision for meshes with arbitrary topology. In: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, pp. 189–192. ACM Press, New York (1996)
24. Georgii, J., Westermann, R.: Mass-spring systems on the GPU. Simulation Modelling Practice and Theory 13, 693–702 (2005)
25. Reddy, J.: An Introduction to Nonlinear Finite Element Analysis. Oxford University Press, Oxford (2004)
26. Baraff, D., Witkin, A.: Physically Based Modelling. ACM SIGGRAPH Course Notes (2003)
27. Bathe, K.: Finite element procedures in engineering analysis. Prentice-Hall, Englewood Cliffs (1982)
28. Hughes, T.: The Finite Element Method: Linear Static and Dynamic Finite Element Analysis. Courier Dover Publications (2000)

Improved Meshless Deformation Techniques for Plausible Interactive Soft Object Simulations

Alex Henriques and Burkhard Wünsche

University of Auckland, Dept. of Computer Science, Graphics Group, Private Bag 92019
Auckland, New Zealand
ahen045,burkhard4u@gmail.com
http://www.cs.auckland.ac.nz/~bwue001

Abstract. Meshless deformation based on shape matching is a new technique for simulating deformable objects without requiring mesh connectivity information. The approach focuses on speed, ease of use and stability at the expense of physical accuracy. In this paper we introduce improvements to the technique that increase physical realism and make it more suitable for use in interactive real-time environments such as games and virtual surgery applications. We also present intuitive real-time interaction techniques for picking, pushing and cutting objects simulated using meshless deformation based on shape matching. For deformable collision detection and response, we present a new method for surface meshes based on previous volumetric methods.

Keywords: Deformable models, real-time simulation, interaction techniques, shape matching, virtual environments.

1 Introduction

Advances in graphics hardware and rendering techniques have made real-time interactive virtual environments increasingly realistic. In the past few years such applications and in particular computer games have started to incorporate rigid-body physics, which are easily controlled and readily simulated using efficient libraries like ODE [12]. As processing power increases further and physics cards are introduced, the natural progression is to include real-time deformable object simulation into virtual environments.

Existing solutions can be divided into pre-animations, kinematic methods, geometric methods, and physically-based methods. Pre-animated simulations are achieved by modelling a limited range of interactions using a human animator, motion capture, or more complex physically-based techniques. The simulations are stored in movie or 3D animation formats and are triggered when the user performs certain predefined operations such as cutting in a specified region. This type of simulation does not allow arbitrary interactions, but is fast and can be achieved using game engines, flash animations and other widely available tools.

Kinematic methods do not represent material properties and forces and include direct mesh manipulation and implicit surfaces. Free-form deformation associates object coordinates with locations in a surrounding mesh of control points [11]. If the control

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 88–101, 2008.

mesh is deformed the object deforms with it. The technique is quite simple, but offers limited forms of manipulations, and makes it difficult to implement cutting operations.

We use the term geometric methods for techniques which use physical properties to kinematically deform regions of an object's geometry. Delp et al. [4] represent different types of tissue using polygonal surface meshes. When interacting with the tissue, the nearest contact point to the surgical instrument is calculated. Affected vertices in a predefined area around this contact point are deformed using a polynomial interpolation.

Physically-based methods include mass-spring systems and finite element methods. Mass-spring systems represent soft objects as a set of points where neighbouring points are connected by springs which simulate the elasticity of the material. The method is easy to implement and cutting can be modeled by removing springs [8]. Finite element simulations model the volumetric nature of soft objects and describe its deformation behaviour using a set of differential equations incorporating material parameters [2]. The resulting simulations are physically realistic but are computationally expensive.

While the above presented methods have been continuously improved in recent years, they are still either computationally expensive or only offer a limited number of interactions and are difficult to implement and integrate into commonly available graphics engines. "Meshless deformation based on shape matching"', or *meshless deformation* for short, is a new technique for simulating deformable objects [9]. The technique is fast, easy to use, unconditionally stable, and has low memory requirements. These factors make the technique particularly interesting for virtual surgery applications and highly interactive real-time environments like computer games.

In this paper we present improvements to this technique. Section 2 introduces the meshless deformation technique in more detail, while section 3 details our improvements to the technique. Section 4 describes the interaction techniques available in the application we have developed, and section 5 describes the collision detection and response methods we implemented. Finally, section 6 summarises our results, and section 7 concludes.

2 Meshless Deformation

In meshless deformation, each object is represented by a set of points, or *point cloud*. No connectivity information is required. Each point in the point cloud moves and responds to forces independently of other points, while meshless deformation ensures the object retains its overall shape. Let the initial configuration (i.e. positions) of points be $\mathbf{x}_i^0$, and the deformed configuration of points at some later time be $\mathbf{x}_i$. To preserve the object's shape, meshless deformation moves and rotates the initial shape $\mathbf{x}_i^0$ as closely as possible onto the actual shape $\mathbf{x}_i$ (see figure 1). The translated and rotated initial shape now defines the set of *goal positions* $\mathbf{g}_i$. Every timestep, each point is moved a fraction α of the way towards its goal position. This gives the point cloud a tendency to preserve its initial shape.

The optimal transformation from $\mathbf{x}_i^0$ to $\mathbf{g}_i$ minimises the sum of the squared distances between $\mathbf{g}_i$ and $\mathbf{x}_i$. The problem is the same as that of "absolute orientation": given coordinates of a set of points as measured in two different Cartesian coordinate systems,

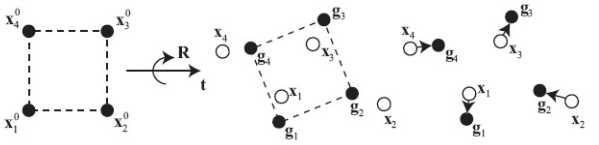


Fig. 1. First, the original shape $\mathbf{x}_i^0$ is matched to the deformed shape $\mathbf{x}_i$. Then, the deformed points $\mathbf{x}_i$ are pulled towards the matched shape $\mathbf{g}_i$ (adapted from [9]).

find the optimal transformation between them [7]. This corresponds to minimizing the following sum.

$$\sum_i w_i \left(\mathbf{R}(\mathbf{x}_i^0 - \mathbf{t}_0) + \mathbf{t} - \mathbf{x}_i\right)^2$$

where $\mathbf{R}$ is a pure rotation matrix. In meshless deformation, the weights are the point masses, $\mathbf{t}_0$ is the centre of mass of the initial shape, and $\mathbf{t}$ is the centre of mass of the current shape. This equation can be extended to allow linear and quadratic matching by replacing $\mathbf{R}$ with a linear deformation matrix $\mathbf{A}$ or quadratic deformation matrix $\widetilde{\mathbf{A}}$. Linear deformations allow stretch and shear, while quadratic deformations additionally allow bends and twists. A tendency towards the undeformed state is introduced by combining $\mathbf{A}$ or $\widetilde{\mathbf{A}}$ with $\mathbf{R}$, resulting in a final deformation matrix $\mathbf{F}$.

$$\mathbf{F} = \beta\widetilde{\mathbf{A}} + (1-\beta)\mathbf{R} \tag{1}$$

where β is a user defined constant between 0 and 1. When β is low, the tendency is largely towards a rigid undeformed state; when β is high, the tendency is more towards the quadratic match, resulting in a softer more deformable object.

2.1 Clusters

Because meshless deformation matches a quadratically deformed version of the initial object, deformation is limited to combinations of stretch, shear, bend and twist over the entire object. This means local deformations – those deforming only one part of an object – are impossible. Higher order deformations, e.g. the cubic deformation of a string given two bends, are also impossible.

As a partial solution to these limitations, Müller et al. divide the set of particles into overlapping clusters with separate deformation matrices. This can greatly increase the range of deformation. However, applications are largely limited to objects with mostly independent subparts that deform only quadratically.

3 Improvements

3.1 Surface Area Preservation

Meshless deformation matches a goal configuration to the deformed point cloud as closely as possible. However, the goal configuration frequently has a different volume than the original object, which is generally undesirable. To preserve volume, meshless

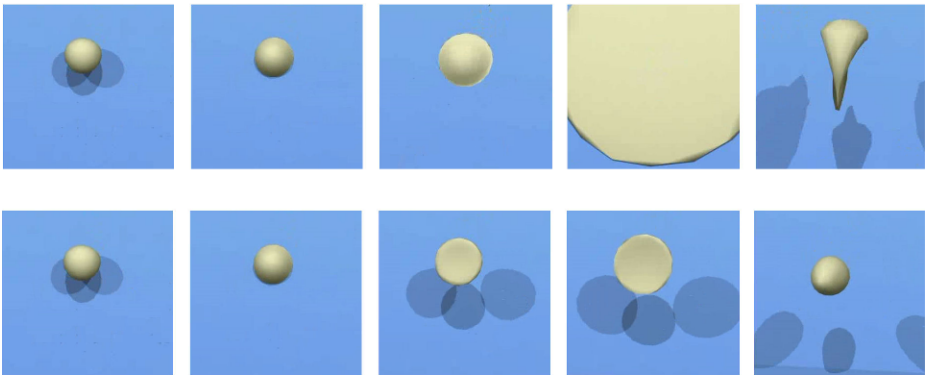


Fig. 2. A ball smashed with high force against a wall. The original meshless deformation algorithm (top) is unconditionally stable but leads to unrealistic intermediate configurations. Our modification (bottom) limits the scale and shear factors and results in more realistic looking results.

deformation scales the deformation matrix such that the goal configuration's volume is identical to the original object's volume. The problem with such "blind" scaling is that when for example a force squashes the object along one dimension, the other two dimensions are scaled up drastically in response as illustrated in the top row of figure 2.

Suggested Solutions. If an airtight balloon filled with water were thrown gently at a wall, the volume of water inside would remain constant. But the balloon would not behave as in the top row of figure 2, because of resistance to *surface area* stretch. Clearly then, a method to constrain surface area is needed. Some algorithms use mesh-based explicit surface area preserving forces [14]. For meshless deformation, possible solutions include the following:

1. Limit forces applied to objects. If the vertices are not subject to large forces, they will not move so far out of their original configuration that blind volume-preservation scaling will produce such extreme surface area changes.
2. Limit the maximum velocities of vertices. As with 1, this will make the occurrence of extreme configurations much less likely.
3. Limit α and β. If α is large, the vertices will return quickly to their goal positions, lessening the likelihood of extreme configurations being produced. If β is small, the tendency of the cube to return to an undeformed state will override the quadratic transformation if it matches an extreme configuration.
4. Have vertices propagate a constraint force through to adjacent vertices.
5. Limit the transformation matrix so that it doesn't match extreme configurations.

1, 2, and 3 used in various combinations are quite successful in combating this problem. 4 is an interesting option, but would require connectivity information to be implemented efficiently. These methods also require tightly regulated parameters, so by definition

cannot be unconditionally stable. 5 on the other hand is simple to implement, efficient, and can achieve unconditional stability.

The simplest way to constrain surface area using 5 is to cap the Frobenius norm of the linear deformation matrix **A**.

$$\|\mathbf{A}\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2$$

Drastic increases in surface area are caused by large stretch or shear values, which are the contributors to $\|\mathbf{A}\|_F^2$. Therefore, by limiting $\|\mathbf{A}\|_F^2$, we limit stretch and shear. If we want to cap the amount of quadratic deformation for visual reasons, the following methods can also be trivially extended from **A** to $\widetilde{\mathbf{A}}$. In the subsequent computations, we use the term $\|\mathbf{A}\|$ as shorthand for the Frobenius norm.

Methods of Clamping. There are several ways to clamp $\|\mathbf{A}\|$; here are three.

1. Let rows of **A** be termed $\mathbf{r}_i$. If any $\|\mathbf{r}_i\|^2$ exceeds a user selected c_{max}, scale $\mathbf{r}_i$ by x such that $\|x\mathbf{r}_i\|^2 = c_{max}$.
2. Cap the magnitude of **A** at c_{max}. To do this, if $\|\mathbf{A}\|^2 > c_{max}$ update **A** as

   $$\mathbf{A} \leftarrow \gamma\mathbf{A} + (1-\gamma)\mathbf{R}$$

   where **R** is the rotation matrix from equation 1 and γ is derived from the solution to the quadratic equation

   $$\|\gamma\mathbf{A} + (1-\gamma)\mathbf{R}\|^2 = c_{max}.$$

   Note that because of the choice of c_{max} the function is monotonically increasing when $0 \le \gamma \le 1$ and hence the quadratic equation has exactly one solution. The final matrix **F** in equation 1 is then calculated as:

   $$\begin{aligned}\mathbf{F} &= (\gamma\mathbf{A} + (1-\gamma)\mathbf{R})\beta + \mathbf{R}(1-\beta)\\ &= \gamma\beta\mathbf{A} + \beta\mathbf{R} - \gamma\beta\mathbf{R} + \mathbf{R} - \beta\mathbf{R}\\ &= \gamma\beta\mathbf{A} + (1-\gamma\beta)\mathbf{R}.\end{aligned}$$

   hence γ is a simple beta modifier, i.e., it makes the deformation more rigid.
3. As a cheaper imitation of 2, simply set

   $$\gamma = \frac{c_{max}}{\|\mathbf{A}\|^2}.$$

The first method works well, but restricts deformation along each axis regardless of deformation in the other axes. The second and third methods on the other hand restrict the sum of deformations along all axes, so maximum deformation along one axis prevents further deformation along the other axes. The appropriate method would seem to depend on the physical properties of the object. Visually we could not distinguish between methods 2 and 3.

Further Extensions. These three methods solve the blow-up problem well, but introduce a slight problem with visual plausibility. A soft object falling to the ground will flatten to the point where the deformation magnitude ϕ is capped, then deformation will jerk to a stop. To solve this we suggest a "soft" cap in the form of a monotonically increasing function f such that for an intermediate threshold c and a maximum threshold m,

$$f(\phi) = \begin{cases} \phi & \phi \leq c \\ < m & \phi > c \end{cases}$$

In other words, the more the deformation magnitude ϕ exceeds the soft cap c, the more it will be reduced such that it never exceeds the hard cap m. Here is an example function:

$$f(\phi) = \begin{cases} \phi & \phi \leq c \\ m - \left(\frac{c}{\phi}\right)(m-c) & \phi > c. \end{cases}$$

3.2 Inversion

Recall that the central equation to be minimised in meshless deformation is

$$\sum_i w_i \left(\mathbf{R}(\mathbf{x}_i^0 - \mathbf{t}_0) + \mathbf{t} - \mathbf{x}_i\right)^2$$

Müller et al. present the most referenced solution to this problem (referred to as that of absolute orientation) derived by Horn [7]. In this paper, however, Horn mentions that the **R** obtained may be a reflection, rather than a rotation, in cases where reflection provides a better fit.

In traditional photogrammetric applications of the absolute orientation problem, the data may seldom be corrupted enough to produce a reflective **R**. When applied to physical objects undergoing large deformations, however, our experiments showed that the vertices frequently are deformed enough that the optimal **R** is a reflection. The inverted object produced is an unacceptable result for homogeneous objects, because it would require massive self-penetration.

Determinant Cube Root Solution. Müller et al. do not specifically mention what to do when a reflective **R** is produced. The only related comment is made when discussing volume preservation of the linear transformation matrix **A**:

> To make sure that volume is conserved, we divide **A** by $\sqrt[3]{det(\mathbf{A})}$ ensuring that $det(\mathbf{A}) = 1$.

When $det(\mathbf{A})$ is negative, $\sqrt[3]{det(\mathbf{A})}$ is also negative. The subsequent division results in an **A** that produces a non-inverted, volume preserving goal position configuration. This configuration is obtained however by a simple reflection of each optimal position through the origin. **A** no longer describes a minimization of goal position with respect to vertex position. Thus the goal positions will tend to be far away from their respective vertex positions, and the integration step will produce large velocities. The result is a blowup.

When taken literally, the method deals with an inverted goal match by producing a blowup. If $\sqrt[3]{det(\mathbf{A})}$ is constrained to its absolute value, the method results in a stable, inverted object configuration. Neither result is acceptable.

Modified R Extraction Solution. Rather than make a modification to the transformation matrix after **R** has been calculated, a modified algorithm is proposed by Umeyama [16] that strictly produces an optimal rotation matrix **R**. Implementing this modification involves only a simple addition to the singular value decomposition solution method of Arun et al. [1].

This method solves the inversion problem, but only partially. While **R** will always be a rotation, **A** may still contain a reflection (assuming the absolute value of $\sqrt[3]{det(\mathbf{A})}$ is used). The final transformation matrix $\mathbf{F} = \beta\mathbf{A} + (1-\beta)\mathbf{R}$ then will always have a tendency towards a non-inverted configuration. But with β close to 1, the tendency will be slow, and may produce physically implausible results. Ideally **A** would calculated in a manner that never produced reflections–this remains for future work.

4 Interaction Techniques

In order to make a virtual world more realistic it is necessary to enable the user to interact with objects in a believable manner. Simulating both the look and feel of materials increases realism and the immersive experience. Furthermore advanced interactions are required for many applications such as virtual surgery simulations. In this section we introduce techniques for picking, constraining, pushing and cutting objects simulated using meshless deformation based on shape matching.

The picking mode allows the user to grab and manipulate any object vertex with a spring force. The spring force acts towards the cursor position (represented by a red sphere), and can also be moved back and forth along the camera's look direction using the mouse wheel. Spring forces can be locked in place, allowing the user to change modes or create new spring forces. In this manner objects can be moved around, bend, and "fixed" in deformed positions (see figure 3). This mode is useful for precisely manipulating an object's position, deformation and orientation.

The pushing mode allows the user to manipulate objects with a pushing force. The cursor position is represented in 3D space as in the picking mode, and collision response forces are applied to any objects near the cursor. This mode is useful for moving several objects at once, as when clearing a path or area.

4.1 Cutting

The main function of the cutting mode is to cut objects into separate pieces. The cursor turns into two cylinders designed to mimic a cutting instrument, e.g. a pair of scissors. To cut an object, the user moves the "scissors" to the appropriate position relative to the object, then holds down the left mouse button to begin the cutting process. The two "blades" of the scissors move closer together, and when they meet, every object the scissors intersect is severed along the plane of the scissors, creating two new separate objects.

Fig. 3. A deformed model of a trout fixed using two locked pick points (left) and a torus model violently moved around using the picking mode (right)

To change the orientation of the scissors, the user can move the scissors towards and away from the view point using the mouse wheel. The scissors can be rotated about the y-axis by holding down shift and pressing the left mouse button.

Our implementation splits an object in two along a plane by using two clipping operations and removing parts outside the clipping plane. If the clipped triangle is a quadrilateral it is divided into two triangles. Note that the cutting plane is derived from the orientation of the cutting tool used in the application. Hence we do not have to deal with partial cuts and the internal surface revealed by the cuts is always planar.

The next step is to seal the exposed cross-sections of the divided object. New surfaces are created by applying a Delaunay triangulation to the newly created vertices touching the cutting plane (see figure 4). The triangles tend to be irregularly shaped because only vertices around the edge of the surface are fed into the algorithm. With no vertices in the centre, each triangle needs to span edge to edge. An improvement to our method would add new vertices inside the edges before running the Delaunay triangulation algorithm, resulting in more consistently sized and shaped triangles. After the triangulation is performed, the object is tetrahedralised and divided into clusters again.

5 Collision

Several types of methods are available for detecting and responding to collisions between deformable objects. These include bounded volume hierarchies, stochastic methods, distance fields, spatial subdivision, and image-space techniques [15].

The collision detection and response techniques used by Müller et al. [9] involve spatial hashing [13] and penetration depth estimation [5] on tetrahedral meshes:

1. Using spatial hashing classify all points intersecting a tetrahedron as colliding.
2. Colliding points connected by an edge to a non-colliding point are *border points*.
3. For each border point a penetration depth and direction is calculated based on connected edges' *intersection points* and corresponding surface normals.
4. Penetration depths and directions are propagated inwards to the remaining colliding points in a breadth-first manner.

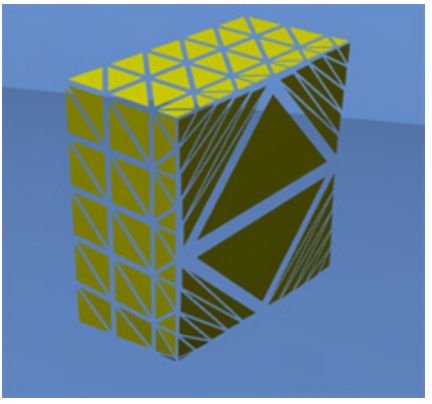


Fig. 4. After a cut, the exposed internal hole is sealed up with a delaunay triangulation

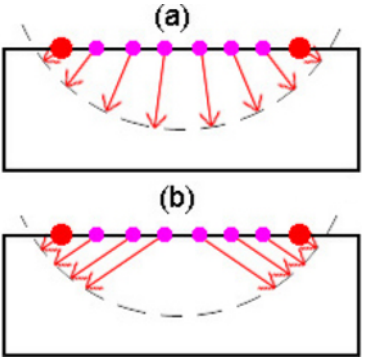


Fig. 5. Response forces: ideal (a) and using a penetration depth estimate

One major disadvantage of this method is that it requires a tetrahedral mesh. Many applications, for example games, use only surface meshes. With this in mind we adapted the method for use with surface meshes.

In steps 1 and 2, we need to classify colliding and border points without tetrahedrons. Using spatial hashing, we test each edge for intersection with nearby surface mesh triangles. On intersection, we classify the edge point in the triangle's positive halfspace as non-colliding, and the edge point in the triangle's negative halfspace as colliding. We also record the length along the edge of the intersection point. If the same edge intersects multiple triangles, the two edge points' classifications are with respect to their closest triangle along the edge. The colliding points so classified are the border points. Remaining points are classified as colliding if they can be reached from a border point without passing through a non-colliding point.

Step 3 remains the same. Step 4 requires significant modification however – figure 5b shows the penetration depths and directions calculated without modification. The problem here is that without a tetrahedral mesh, border points only exist on the surface of the mesh around the intersecting triangles, and not inside the mesh around deeply penetrated areas. This results in unrealistic propagated penetration directions. Rather than use propagation, for each non-border colliding point we simply calculate the penetration direction as a weighted sum of each border point's penetration direction, where the weights are inversely proportional to the number of edges in the shortest edge path between the colliding point and border point. To calculate penetration depth, we find the length of a ray cast from the colliding point to the surface along the penetration direction. The results are as in figure 5a.

Compared to the original tetrahedral method, our surface mesh collision technique is slower and subject to classification errors and erratic behaviour. While the method works for simple applications further research is necessary to make it more stable and hence suitable for computer games and similar applications where tetrahedral meshes are not available.

6 Results

We have developed a framework for testing interactive simulation environments and implemented within it meshless deformation based on shape matching together with our improvements. The user may pick, push or cut deformable objects in real-time.

6.1 Simulation Capabilities

In order to determine the suitability of meshless deformation for interactive simulations we tested it on a variety of objects using different deformations:

Simulation of Jelly-like Cubes and Spheres. The four basic modes of deformation possible are stretch, shear, bend and twist. We tested the real-time rigid, linear, and quadratic deformation of a deformable cube and sphere subjected to user interaction. Visual plausibility was good; the objects behaved as one would expect.

Simulation of a Soft L-shaped Bar. This test investigates the deformation behaviour for large displacements applied to highly concave objects. We would expect to be able to bend both sections of the bar together or apart, but using only one cluster this is not possible as illustrated in figure 6 (a)-(b). In particular even with quadratic behaviour the bar twists and bounces strangely (c). A significant improvement is achieved by using one cluster for each branch of the L-bar. Results are plausible (see figure 7), however where clusters meet at the corner of the bar, deformation is uneven when the bar is pulled straight (c).

Simulation of Complex Objects. We tested a variety of complex objects and found that objects with limited bending modes, such as a trout, are simulated well (left image of figure 3). It seems that the quadratic deformation modes of the trout model (which is a surface model) correspond well with the type and magnitude of deformations of a real trout which are restricted by its rigid skeletal structure. Surprisingly also some very complicated objects such as an intertwined rubber torus look realistic (right image of figure 3). The main reason for this seems to be that users are not familiar with its behaviour. An informal user survey with students revealed that there was no consensus how the object should behave when deformed. In contrast we are intimately familiar with the deformations of a human face and any deviation from physical accuracy can be easily noticed. We also found that to achieve an acceptable range of deformations corresponding to the muscle groups of the face, clusters needed to be divided very precisely - we had to implement a special export tool to allow precise cluster specification in a 3D modeling program. Even then, we found clusters very difficult to manipulate into giving plausible facial animations, and boundaries between clusters were often noticeable.

Simulation of Soft Tissue. For the final test we tested the suitability of meshless deformation for virtual surgery simulations [6]. Good results were obtained when applying large deformations to blobby objects such as kidney shaped models and convoluted tube like structures. A trained user could easily notice that the deformations were physically

not realistic, however, when using clusters they were physically plausible and sufficient for simulating process related surgical tasks.

Modelling local deformations, e.g., for a section of skin, is difficult. Figure 8 (a) shows that for a skin patch consisting of one cluster ($\alpha = 1.0$, $\beta = 0.6$), picking simply attempts to move and deform the entire skin patch. When the skin patch is divided into 2×2 clusters deformation is more plausible but still limited, with the dividing lines between clusters quite visible in figure 8 (b)-(d). Using 5×5 clusters significantly increases visual plausibility to a satisfactory level (figure 9). Using such a large number of clusters is, however, quite inefficient and less accurate than using a mass-spring system with a similar resolution.

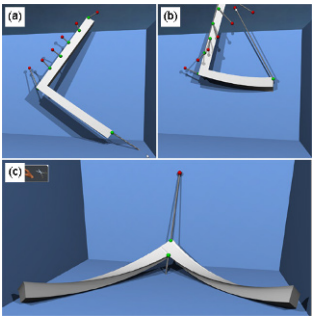


Fig. 6. L-bar without clusters being pulled apart

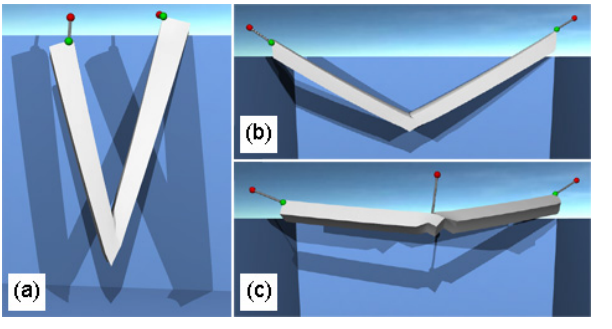


Fig. 7. The same operation for an L-bar modelled with two clusters

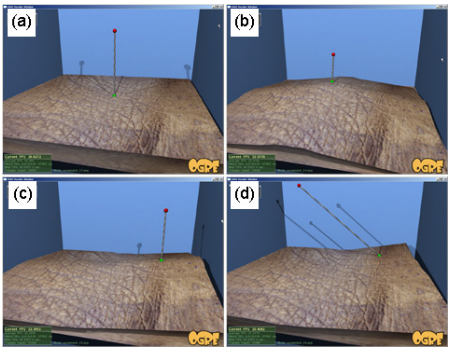


Fig. 8. (a) pick on a single cluster skin patch, (b)-(d) picks on a 2×2 cluster skin patch

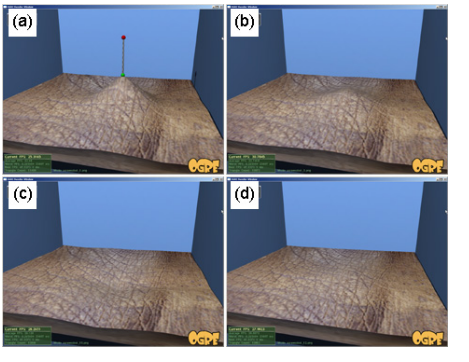


Fig. 9. Behaviour of a 5×5 cluster skin patch in response to a user pick

In summary we found that simple objects with limited modes of deformation are simulated best, while objects composed of simple subcomponents are simulated well with clusters. Objects with a very high number of deformation modes, such as cloth, can not be simulated efficiently [10]. However, the method seems to work well for complicated objects which have simple deformation modes (e.g., a trout) or with where the user is unfamiliar with its behaviour (e.g., an intertwined rubber torus).

Local deformations can not be modelled efficiently and are best approximated by clusters. An alternative solution are hybrid models combining meshless deformation and, for example, a mesh spring model. Such multi-resolution representation where non-linear responses are only considered in the immediate vicinity of the applied force in order to obtain real-time non-linear deformations exist already, however, they require model representation which are not suitable for game engines and other common graphics engines for virtual environments [3].

6.2 Suitability for Real-Time Collaborative Interactive Environments

We identified a range of criteria which a soft object simulation technique for real-time interactive collaborative environments, such as computer games or Virtual Worlds, must fulfill.

Usability. Informal user testing indicates that our environment and all our interaction techniques were intuitive and easy to use. The ability to push, pull, fix and cut deformable colliding objects significantly increased user enjoyment. In particular the cutting tool proofed surprisingly popular in our informal user testing (figure 10).

Stability. Due to the improvements implemented, extreme forces and deformations no longer produce stable inversions or erratic behaviour due to temporary inversions. Furthermore, large forces no longer result in surface area blowups, allowing the use of arbitrary stiffness (α and β) values, forces and speeds. Objects that are particularly soft or moving at great speeds no longer jerk to a sudden stop when their deformations exceed a certain amount, instead gradually reaching a maximum deformation between soft and hard caps.

Ease of implementation. We found meshless deformation relatively easy to implement and integrate into the 3D rendering engine Ogre. There are only two main differences between current 3D engines and what is required for deformable object simulation. Firstly, rigid objects have static sharable meshes, while deformable objects require updates to individual vertex positions every timestep on their own mesh instance. Secondly, collision detection and response is a much slower, more difficult task for deformable objects.

Performance. Our environment is comparatively fast: We can simulate dozens of simple 32 tetrahedron objects with collisions in real-time and unconditional stability (see figure 11). Higher speeds, e.g., for simple virtual surgery applications, could be achieved by optimising our algorithms and/or implementing them on the GPU.

Tweakability. The "gooeyness" and stiffness of each object can be easily modified using the α and β parameters. Further collision-response parameters can also be tweaked. The strength of surface area preservation can be specified with a force response curve. Volume preservation is automatic, but can be adapted to use a force response curve as well.

Disadvantages. The primary disadvantage of our environment is the lack of robust local deformation. For complex applications which require plausible localised deformation of an arbitrary region, e.g., motor skill training of surgeons, our environment is less suitable. Also, even when the simulation is visually plausible, it is usually not physically accurate.

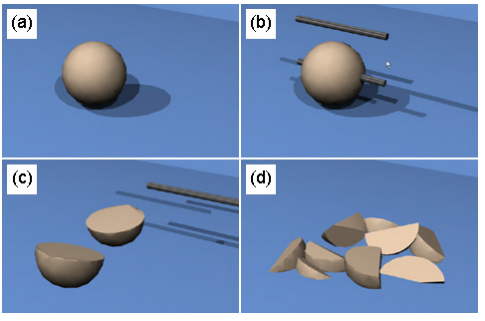


Fig. 10. Cutting an object: (a) during cut, (b) immediately after cut, (c) two resulting halves have rolled apart, (d) after further cuts

Fig. 11. Large scale simulation of deformable objects

7 Conclusions

We have implemented an improved algorithm for meshless deformation based on shape matching. Our improvements include soft capped surface area preservation, and the prevention of inverted states. We have also implemented several techniques enabling users to interact with deformable objects realistically and intuitively. Collision detection and response have been implemented based on spatial hashing and accurate penetration depth estimation techniques. We have also adapted the collision method for use with triangular surface meshes, for applications such as games where tetrahedral meshes are not available. Informal user testing indicates that users find our environment significantly more enjoyable and immersive than a comparable rigid body physics environment.

Disadvantages include that simulating local deformations requires division of the object into fine grained clusters, which can be inefficient. Precise cluster divisions can also be difficult to specify. For large scale objects and scenes, efficiency improvements are necessary. Finally, the cut operation does not support partial cuts or incisions, which would be useful for virtual surgery applications or games.

In summary, we believe that the techniques implemented have promising potential as applied to a virtual surgery simulator, games, or any other environment where speed and immersive interactions are required but physical accuracy is not.

8 Future Work

One major problem limiting meshless deformation's use in some applications is the lack of robust local deformation. One avenue of investigation might be to integrate a mass-spring system, which is usually disabled, but where user picks activate mass-spring behaviour in the pick's local region. Mass-spring areas around a partial cut or incision could similarly be activated. For larger cuts, but not complete severances, a method of dynamically partitioning new clusters may be possible that would allow "flapping" behaviour, similar to a tennis ball nearly cut in half with both halves "talking" like a mouth.

References

1. Arun, K., Huang, T., Blostein, S.: Least-squares fitting of two 3-D point sets. IEEE Transactions on Pattern Analysis and Machine Intelligence 9(5), 698–700 (1987)
2. Berkley, J., Turkiyyah, G., Berg, D., Ganter, M., Weghorst, S.: Real-time finite element modeling for surgery simulation: An application to virtual suturing. IEEE Transactions on Visualization and Computer Graphics 10(3), 314–325 (2004)
3. De, S., Lim, Y.-J., Manivannan, M., Srinivasan, M.A.: Physically realistic virtual surgery using the point-associated finite field (paff) approach. Presence: Teleoperators and Virtual Environments 15(3), 294–308 (2006)
4. Delp, S.L., Loan, P., Basdogan, C., Rosen, J.M.: Surgical simulation: an emerging technology for emergency medical training. Presence: Teleoperators and Virtual Environments 6(2), 147–159 (1997)
5. Heidelberger, B., Teschner, M., Keiser, R., Muller, M., Gross, M.: Consistent penetration depth estimation for deformable collision response. In: Proceedings of Vision, Modeling, Visualization VMV 2004, Stanford, USA, pp. 339–346 (2004)
6. Henriques, A., Wünsche, B.C., Marks, S.: An investigation of meshless deformation for fast soft tissue simulation in virtual surgery applications. International Journal of Computer Assissted Radiology and Surgery 2 supp.1, 169–171 (2007); Proceedings of the 21st International Congress and Exhibition on Computer Assisted Radiology and Surgery (CARS 2007) (June 2007)
7. Horn, B.: Closed-form solution of absolute orientation using unit quaternions. Journal of the Optical Society of America A 4(4), 629–642 (1987)
8. Kühnapfel, U., Cakmak, H., Maass, H.: Endoscopic surgery training using virtual reality and deformable tissue simulation. Computers & Graphics 24, 671–682 (2000)
9. Müller, M., Heidelberger, B., Teschner, M., Gross, M.: Meshless deformations based on shape matching. ACM Trans. Graph. 24(3), 471–478 (2005)
10. Rubin, J., Wünsche, B.C.: A framework for interactive and physically realistic cloth simulation. 780 project report, University of Auckland (Febuary 2006), `http://www.cs.auckland.ac.nz/ /burkhard/Reports/2005_SS_JonathanRubin.pdf`
11. Sederberg, T.W., Parry, S.R.: Free-form deformation of solid geometric models. ACM Transactions on Graphics 20(4), 151–160 (1986)
12. Smith, R.: Open Dynamics Engine home page (2006), `http://www.ode.org`
13. Teschner, M., Heidelberger, B., Mueller, M., Pomeranets, D., Gross, M.: Optimized spatial hashing for collision detection of deformable objects (2003)
14. Teschner, M., Heidelberger, B., Muller, M., Gross, M.: A versatile and robust model for geometrically complex deformable solids. In: Proceedings on Computer Graphics International, pp. 312–319 (2004)
15. Teschner, M., Kimmerle, S., Zachmann, G., Heidelberger, B., Raghupathi, L., Fuhrmann, A., Cani, M.-P., Faure, F., Magnetat-Thalmann, N., Strasser, W.: Collision detection for deformable objects. In: Eurographics State-of-the-Art Report (EG-STAR) (2004), pp. 119–139. Eurographics Association (2004)
16. Umeyama, S.: Least-squares estimation of transformation parameters between two point patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence 13(4), 376–380 (1991)

Computer Vision Theory and Applications

Part I
Image Formation and Processing

Objective Evaluation of Image Mosaics

Jani Boutellier[1], Olli Silvén[1], Marius Tico[2], and Lassi Korhonen[1]

[1] Machine Vision Group, University of Oulu, P. O. Box 4500, 90014
University of Oulu, Finland
{bow,olli.silven,leevi}@ee.oulu.fi
[2] Nokia Research Center, P.O. Box 100, 33721 Tampere, Finland
tico@ieee.org

Abstract. Image stitching is an image processing method, where multiple photographs covering different parts of the same scene, are combined to form a single wide-angle image. Stitching is a very challenging task, and during the past decades many algorithms have been developed for it. Unfortunately, there has been no objective way to measure the quality of stitching results. To mend this shortcoming, we propose a novel method for testing stitching algorithms. The testing process starts from an arbitrary reference image that is used to create synthetic input data for the stitching algorithm that is to be tested. To make the testing realistic, various camera-related distortions along with perspective warps are applied to the input images. From this input data, the stitching algorithm creates a wide-angle image that is then compared to the reference image, from which the process started.

Keywords: Stitching, mosaics, panoramas, image quality.

1 Introduction

Image stitching is a method for combining several images into one wide-angled mosaic image. Computer-based stitching algorithms and panorama applications have been used widely for more than ten years [1], [2]. Although it is evident that technical improvements have taken place in computer-based image stitching, there has been no objective measure for proving this trend. Subjectively, it is relatively easy to say whether a mosaic image has flaws or not [3], but analyzing the situation computationally is not straightforward at all.

If we assume that we have a mosaic image and wish to evaluate it objectively, the first arising problem is usually the lack of a reference image. And even if we had a reference image of the same scene, we would generally notice that it did not have exactly the same projection as the mosaic image, therefore making pixel-wise comparison impossible. Also, it may happen that between taking the hypothetical reference image and the narrow-angled mosaic image parts, the scene might have changed somewhat, making the comparison unfit.

In this work, a method is described to overcome these problems, but before going deeper into the topic, some terms need to be agreed upon. From here on, the narrow-angle images that are consumed by a stitching algorithm are called *source images*. Also, we will call the group of source images a *sequence*, even if the source images are stored

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 107–117, 2008.

Table 1. Common flaws in image mosaics

Type of Error	Cause
Discontinuity	Failed or inadequate source image registration
Blur	Shooting conditions, unfit blending, lens distortions
Object Clipping	Moving object in source image ignored
Intensity Change	Color balancing between mosaic parts

Fig. 1. Mosaic flaws. From left to right: intensity change, discontinuity, object clipping.

as separate image files. It is worth mentioning that the source images are given to the stitching algorithm in the same order as they have been created.

To be able to create a method of evaluating mosaics, we have to know what kinds of errors exist in mosaic images. Flaws that we will call *discontinuities* are caused by unsuccessful registration of source images. Apart from completely failed registration, the usual cause of these kinds of errors is the use of an inadequate registration method. For example, if the registration method of a stitching algorithm is unable to correct perspective changes of the source images to the mosaic, the mosaic will have noticeable boundaries.

Blur is a common flaw in most imaging occasions and may be caused by the imaging device or by extrinsic causes, e.g. camera motion. In mosaicking, blur can also be caused by inadequate source image blending [4].

Object clipping happens when the location of an object changes in the view of the camera between the source image captures. In a practical image mosaic, a common clipped object is for example a moving pedestrian.

The final mosaic flaw introduced here can only be considered an error in its most dramatic forms: *intensity changes* result from the source frame blending process, when the stitching algorithm balances the brightness of different source images to fade the intensity differences that exist between source images. A summary of mosaic flaws is shown in Table 1 and Figure 1 shows visual examples.

2 Related Work

Since our work was originally published in [5], research has progressed in this field. Paalanen *et al.* [6] used a framework very similar to ours to evaluate the quality of

mosaic images. In addition to using artificial sequences as in our work, Paalanen *et al.* also propose a methodology that uses real images.

Some time ago Marzotto *et al.* [7] developed a method of super-resolution video mosaicking and estimated the quality of the results by measuring the amount of blurring. Clearly this approach is not capable of noticing registration errors. Feldman *et al.* [8] has also considered the quality of image mosaics, but his approach is limited to multi-perspective mosaics.

Swaminathan *et al.* [9] have developed a metric to quantify effects of projective distortions in non-single viewpoint imaging, like mosaicking. This method does not address the question of registration or blending quality, but assumes that the mosaics are created without errors. Furthermore, the application of this method requires some knowledge about the structure of the scene that is depicted in the mosaic. Bors *et al.*[10] have also analyzed perspective distortions, but the method is applied to image sequences that are taken from cylindrical surfaces.

Somewhat related to our problem is also the work that has been done in the field of evaluating the success of image registration. Möller *et al.* [11] have presented a concept for analyzing errors in image registration and Schestowitz *et al.* [12] have developed a method for assessing the performance of non-rigid registration algorithms.

3 Our Approach

Our approach to measure the performance of mosaicking algorithms was motivated by the intention to eliminate all possible sources of disturbances in the quality assessment. To make the testing process as controllable as possible, we have chosen to simulate the generation of source images (*i.e.* the imaging process). Practically this is achieved by using a reference image that acts as real world scenery in the test environment. A sequence of source images is created from the reference image, as if photographing a view by multiple shots that cover different areas of the scenery. For every captured source image, a selected group of common imaging distortions are applied to simulate real-world shooting conditions.

We assume that the stitching algorithms to be tested are completely automatic and produce mosaic images from the sequences that they receive. The generated sequence is handed over to the stitching algorithm that is chosen to be tested. After the mosaicking algorithm has processed the sequence, a mosaic image and a reference image that depict the same view are available. However, there exist changes in intensity and projection

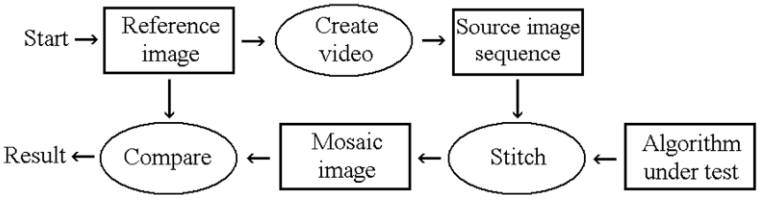


Fig. 2. The process of mosaic quality evaluation

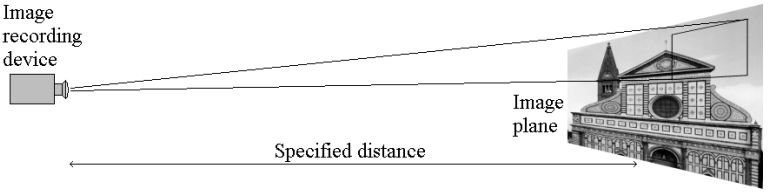


Fig. 3. The simulated imaging environment

between the mosaic and the reference image, as well as possible errors introduced by the stitching algorithm.

To make the pixelwise comparison between the mosaic and the reference image possible, the images need to be brought to the same coordinate frame. This is done by non-rigid image registration. After the registration, the images can be compared with a selected *full-reference* quality assessment algorithm [13]. A block diagram of this process can be seen in Figure 2.

Our approach is only eligible, if the applied image registration algorithm fills two requirements: a) the registration algorithm must be sufficiently powerful to be able to register all error-free mosaic images against the reference image and b) the registration algorithm must be robust enough not to fail even when there are some errors in the mosaic.

The first requirement can be met sufficiently well with some non-rigid registration algorithms when the applied distortions of the imaging process are not too severe. The second requirement is much harder to achieve, since it can happen for example, that the reference image does not provide enough feature points. Because of these requirements, we have to assume that the result of our measurement method can only be as reliable as the non-rigid registration algorithm that it uses.

Related to the registration algorithm, it is also necessary to mention how we define a stitching error: a deformation in the mosaic image is considered an error, if it introduces a new discontinuity that is not present in the reference image. This also means that the applied non-rigid registration method must do the registration by a continuous deformation, not in a piecewise fashion, like block-matching.

3.1 Creating Source Image Sequences

Our method of creating source images can be thought to produce images from a situation, where the camera is at some fixed distance from the reference image that appears as a plane in 3D-space. (See Figure 3). As a result, perspective distortions appear in the source images. In literature this is called the *pinhole camera model* [14].

Images that are normally delivered to a stitching algorithm for mosaicking have several kinds of distortions caused by the camera and imaging process. Real-world camera lenses cause vignetting and radial distortions to the image [14]. If the user takes pictures freehand, the camera shakes and might cause motion blur to the pictures. Also, it is common that the camera is slowly rotated from one frame to another when shooting many pictures of the same scene. Finally, cameras tend to adapt to lighting conditions

by normalizing the global image brightness according to the brightness of the view that is seen through the camera lens.

The frames are created by panning the simulated camera view over the reference image in a zig-zag pattern and by taking shots with a nearly constant interval (see Figure 4). The camera jitter is modeled by random vertical and horizontal deviations from the sweeping pattern along with gradually changing camera rotation. Motion blur caused by camera shaking is simulated by filtering the source image with a point-spread function consisting of a line with random length and direction.

Camera lens vignetting is implemented by multiplying the source image with a two-dimensional mask that causes the image intensity to dim slightly as a function of the distance from the image center. Radial distortions are created by a simple function that is given in equation 1.

$$d_n = d + kd^3, \tag{1}$$

where

d is the distance from the image center,
d_n is the new distance from the center and
k is the distortion strength parameter.

The radial distortion is applied by calculating a new distance for every pixel from the image center in the source image. The result of this warp is a barrel distortion if the constant k is positive. Finally, the simulated differences in exposure time are applied to the source image by normalizing the mean of the image to a constant value.

Figure 5 shows the effect of each step in this simulated imaging process. In the order from left to right and from top to bottom, the distortions are following: perspective-warped part of reference image, barrel-distorted, motion blurred, rotated, cropped result with mean normalized, cropped result with vignetting. Notice that the distortions accumulate from one sub-figure to the next.

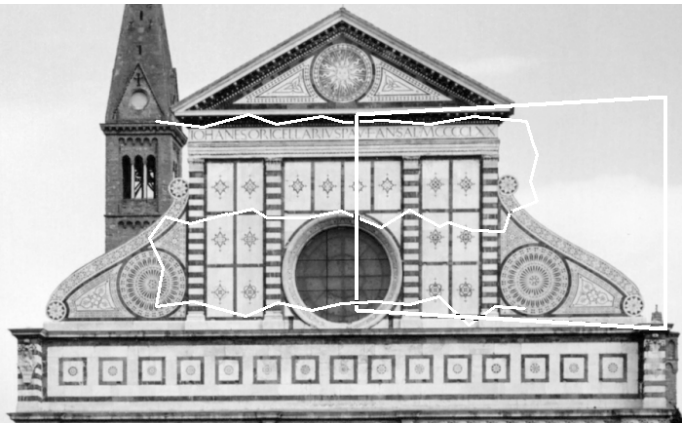


Fig. 4. Camera motion pattern over a reference image and a quadrangle depicting the area included in an arbitrary source image

Fig. 5. The six phases of applying distortions to source images

The source image sequence is recorded as an uncompressed video clip by default, but can of course be converted to other forms depending on the required input type of the algorithm that is chosen for testing.

3.2 Mosaic Image Registration

The stitched mosaic image and the reference image have different projections because the stitching software has had to fit together the source images that contain non-linear distortions. The mosaic image has to be registered to the coordinates of the reference image to make the comparison eligible.

For this purpose we selected a SIFT-based [15] feature detection and -matching algorithm[1], that produced around one thousand matching feature points for each image pair. An initial registration estimate is calculated by a 12-parameter polynomial model, after which definite outliers are removed from the feature point set. The final registration is made by the unwarpJ -algorithm [16] that is based on a B-spline deformation model. It is evident that the success of image registration is a most important factor to ensure the eligibility of the quality measurement. According to the conducted tests, the accuracy and robustness of unwarpJ are suitable for the purpose, although not perfect.

An example of the registration process can be seen in Figure 7. The topmost image is a mosaic image created by a stitching algorithm. The middle image in Figure 7 shows the registered version of the mosaic above, and the image in the bottom shows the corresponding *similarity map* (See next subsection). Notice how slight stitching errors have prevented flawless matching in the bottom-right corner of the image.

[1] http://www.cs.ubc.ca/~lowe/keypoints/

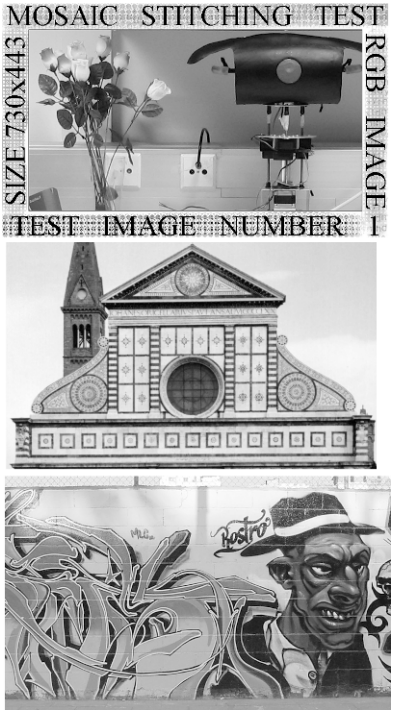


Fig. 6. Used reference images

Fig. 7. Mosaic registration

3.3 Similarity Calculation

Our method uses the recent approach of Wang [13] to estimate the quality of the registered mosaic. The approach of Wang estimates the similarity of two images and gives a single similarity index value that tells how much alike the two images are. This method fits very well to the requirements of mosaic evaluation, since it pays attention on distortions that are clearly visible for the human vision system. This includes blurring and structural changes that are common problems in mosaics. Wang's method does not penalize for slight changes in the image intensity.

This arrangement will notice blurring and discontinuity -flaws in the mosaic images that are created from test videos. However, with the current test setup it is not possible to simulate situations that would cause object clipping.

4 Practical Tests

We used three different stitching algorithms to test the functionality of our test method: Autostitch [17], Surveillance Stitcher [4] and an algorithm that we shall call Mobile Stitcher [18]. The algorithms were tested by three different video sequences that were created from the images shown in Figure 6. Each algorithm was tested with the three

Table 2. Mosaic quality values and visually observed distortions of test mosaics. The numbers in braces indicate which figure displays the corresponding result, if it is shown. DC is an abbreviation discontinuity.

Algorithm	Pattern	Facade	Graffiti
UBC Autostitch	0.81 (7), blur, slight dc	0.89 (8), blur	0.76, blur
Mobile stitcher	0.80 (7), dcs	0.68, dcs	0.65, dcs
Surveillance stitcher	0.75 (7), blur, dc	0.86, blur, slight dc	0.72 (6), blur, dc

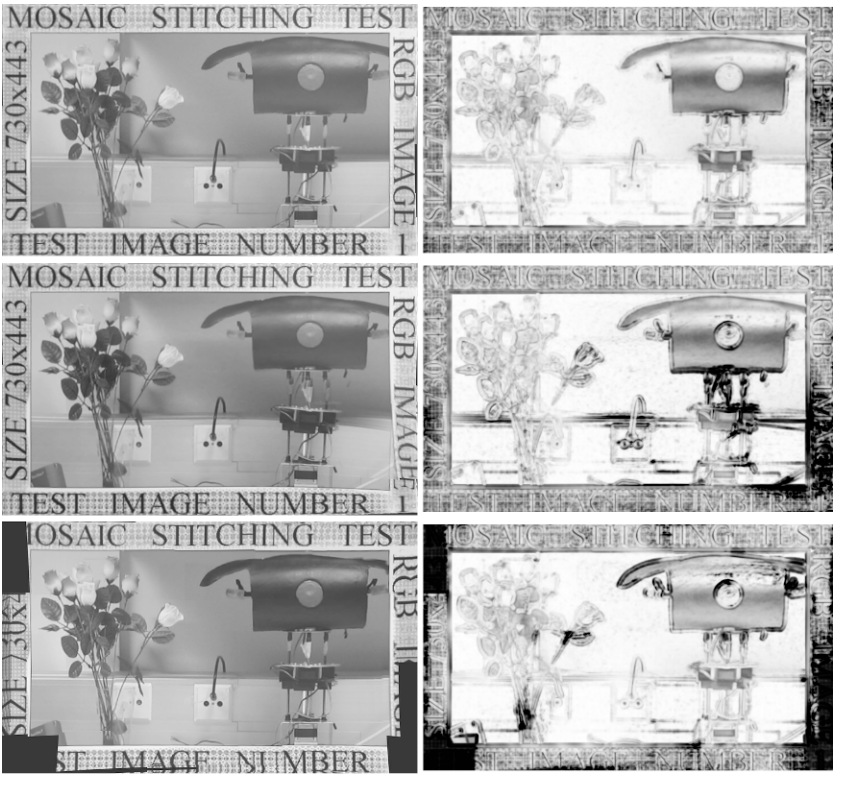


Fig. 8. Mosaics created by three different mosaicking algorithms and the corresponding similarity maps

sequences. Results of subjective mosaic evaluation and quality indexes provided by our algorithm are visible in Table 2.

We shall take a closer look at one example. The left column in Figure 8 shows mosaics created by three different mosaicking algorithms and the right column shows the corresponding similarity maps. From the similarity maps we can see the locations of stitching errors as dark areas. The blur that is present in every mosaic can be seen in the similarity maps as dark object outlines.

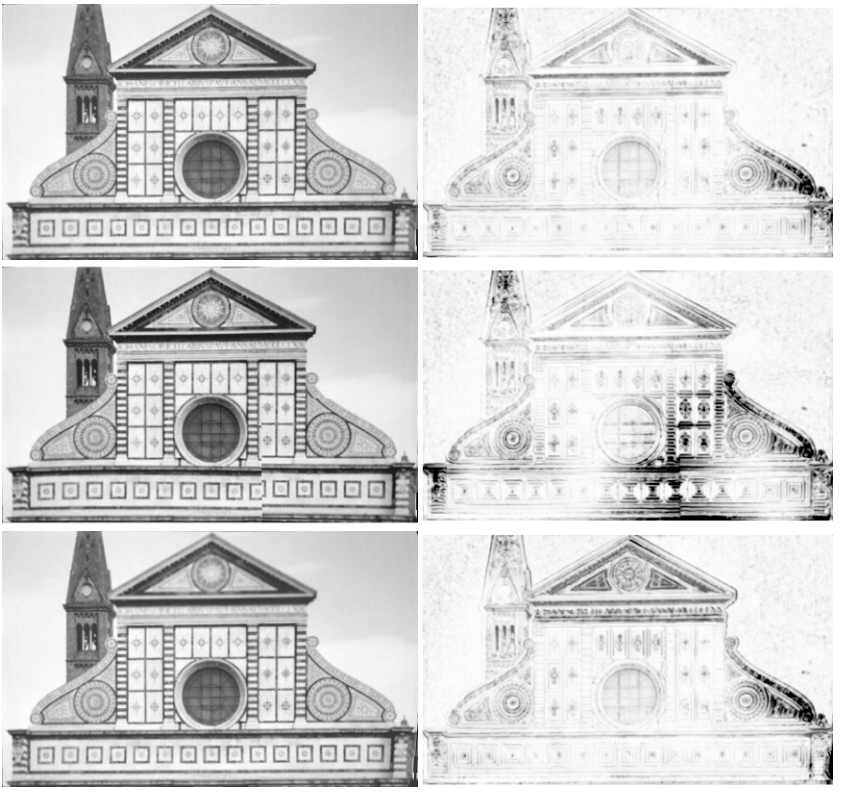


Fig. 9. Effect of stitching errors to the similarity map

The top row shows the results of Autostitch. We can see that the overall quality is all right, yet there is a dark area in the right border. With careful inspection we notice that this is caused by the area near the letter 'B' that is distorted.

The middle row shows the results of the Surveillance Stitcher. It can be easily detected that the dark area near the bottom-right corner is caused by severe discontinuities that also reflect to the areas left of the distortion.

The pictures in the lowest row are produced by the Mobile Stitcher. The Mobile Stitcher is not capable of correcting flaws caused by perspective distortions, and therefore contains problematic areas here and there. For example, below the rightmost flower, there is a discontinuity that is marked as a black area in the similarity map. The same applies for example at the top-right corner that is quite jagged. The Mobile Stitcher has also left considerable holes to the mosaic near the borders.

We can notice from the results that the acquired quality indexes are not comparable from one sequence to another. The focus of this testing was not to sort the tested algorithms to some order of quality, but to simply show what kinds of results can be achieved with our testing method. When the similarity indexes were calculated, 50 pixels from each image border were omitted, since the non-linear registration algorithm was often a bit inaccurate near image borders.

Autostitch acquired the best results from each test, which can also be detected visually, since the results are practically absent of discontinuities. The Surveillance Stitcher acquired second best results, although most of its results had slight discontinuities. The Mobile Stitcher performed worst in these tests, which is easily explained by the fact that the algorithm is the only one of the three that uses an area-based registration [19] method and thus is unable to correct perspective distortions.

Figure 9 shows a mosaic that was created by Autostitch along with some manually damaged versions of the mosaic. The figure depicts how different kinds of mosaicking errors affect the similarity map and the numerical quality of the mosaic. The topmost row shows the already registered mosaic created by Autostitch, which is of good quality (similarity index 0.89).

In the middle row the mosaic created by Autostitch was tampered manually before registration so that a discontinuity appears in the lower half of the image. Then this modified mosaic was registered against the reference image and the similarity map was calculated. As we can see, the discontinuity has made the corresponding area black in the similarity map (similarity index 0.83).

In the lowest row of the figure the roof triangle of the building has been manually blurred before registration. In the similarity map the roof triangle appears darker (similarity index 0.85).

As mentioned earlier, the performance of our comparison approach is heavily dependent on the used image registration algorithm. This came out as the currently used registration method proved to be inaccurate in one occasion. In Figure 9 each similarity map indicates that something is wrong in the right end of the building. However, visual inspection reveals no problems. The reason behind the indicated difference is effectively a misalignment of a few pixels. The misalignment is caused by the lack of feature points in the reference image and a registration error that has followed from this.

5 Conclusions

We have presented a novel way to measure the performance of stitching algorithms. The method is directly applicable to computer-based algorithms that can automatically create mosaic images from source image sequences.

The method could be improved by using depth-varying 3D models, which would test the capabilities of the algorithms to cope with effects of occlusion and parallax. Also, the presence of moving objects could be simulated in future versions of the testing method.

Furthermore, the present similarity computation approach could be replaced with a more sophisticated approach, such as the one presented by Möller [11].

References

1. Davis, J.: Mosaics of scenes with moving objects. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 354–360 (1998)
2. Szeliski, R.: Image Mosaicing for Tele-Reality Applications. DEC Cambridge Research Lab Technical Report 94/2 (1994)

3. Su, M.S., Hwang, W.L., Cheng, K.Y.: Analysis on multiresolution mosaic images. IEEE Transactions on Image Processing 13, 952–959 (2004)
4. Heikkilä, M., Pietikäinen, M.: An image mosaicing module for wide-area surveillance. In: Proceedings of the ACM Workshop on Video Surveillance and Sensor Networks, pp. 11–18 (2005)
5. Boutellier, J., Silvén, O., Korhonen, L., Tico, M.: Evaluating stitching quality. In: Proceedings of VISAPP 2007, pp. 10–17 (2007)
6. Paalanen, P., Kämärainen, J.K., Kälviäinen, H.: Image Based Quantitative Mosaic Evaluation with Artificial Video. Lappeenranta University of Technology, Research Report 106 (2007)
7. Marzotto, R., Fusiello, A., Murino, V.: High resolution video mosaicing with global alignment. In: Proceedings of the IEEE Conference of Computer Vision and Pattern Recognition, vol. 1, pp. 692–698 (2004)
8. Feldman, D., Zomet, A.: Generating mosaics with minimum distortions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop, pp. 163–170 (2004)
9. Swaminathan, R., Grossberg, M.D., Nayar, S.K.: A perspective on distortions. In: Proceedings of the IEEE Conference of Computer Vision and Pattern Recognition, vol. 2, pp. 594–601 (2003)
10. Bors, A.G., Puech, W., Pitas, I., Chassery, J.M.: Perspective distortion analysis for mosaicing images painted on cylindrical surfaces. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 4, pp. 3049–3052 (1997)
11. Möller, B., Posch, S.: An integrated analysis concept for errors in image registration. In: Proceedings of 7th Open German/Russian Workshop on Pattern Recognition and Image Understanding, Ettlingen, Germany (2007)
12. Schestowitz, R., Twining, C., Cootes, T., Petrovic, V., Taylor, C., Crum, W.: Assessing the accuracy of non-rigid registration with and without ground truth. In: 3rd IEEE International Symposium on Biomedical Imaging: Nano to Macro, pp. 836–839 (2006)
13. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing 13, 600–612 (2004)
14. Jähne, B.: Digital Image Processing, Concepts, Algorithms, and Scientific Applications, 4th edn. Springer, Berlin (1997)
15. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60, 91–110 (2004)
16. Sorzano, C.O., Thevenaz, P., Unser, M.: Elastic registration of biological images using vector-spline regularization. IEEE Transactions on Biomedical Engineering 52, 652–663 (2005)
17. Brown, M., Lowe, D.G.: Recognising panoramas. In: Proceedings of the 9th International Conference on Computer Vision, vol. 2, pp. 1218–1225 (2003)
18. Boutellier, J., Bordallo-Lopez, M., Silvén, O., Tico, M., Vehviläinen, M.: Creating panoramas on mobile phones. In: Proceeding of SPIE Electronic Imaging 2007, San Jose, California, USA, vol. 6498 (2007)
19. Zitová, B., Flusser, J.: Image registration methods: a survey. Image and Vision Computing 21, 977–1000 (2003)

Part II
Image Analysis

A Revisited Half-Quadratic Approach for Simultaneous Robust Fitting of Multiple Curves

Jean-Philippe Tarel[1], Pierre Charbonnier[2], and Sio-Song Ieng[1]

[1] ESE, Laboratoire Central des Ponts et Chaussées, 58 Bd Lefebvre, 75015 Paris, France
Tarel@lcpc.fr, Sio-Song.Ieng@lcpc.fr
http://perso.lcpc.fr/tarel.jean-philippe/
[2] ERA 27 LCPC, Laboratoire des Ponts et Chaussées, 11 rue Jean Mentelin
B.P. 9, 67035 Strasbourg, France
Pierre.Charbonnier@equipement.gouv.fr

Abstract. In this paper, we address the problem of robustly recovering several instances of a curve model from a single noisy data set with outliers. Using M-estimators revisited in a Lagrangian formalism, we derive an algorithm that we call Simultaneous Multiple Robust Fitting (SMRF), which extends the classical Iterative Reweighted Least Squares algorithm (IRLS). Compared to the IRLS, it features an extra probability ratio, which is classical in clustering algorithms, in the expression of the weights. Potential numerical issues are tackled by banning zero probabilities in the computation of the weights and by introducing a Gaussian prior on curves coefficients. Applications to camera calibration and lane-markings tracking show the effectiveness of the SMRF algorithm, which outperforms classical Gaussian mixture model algorithms in the presence of outliers.

Keywords: Image Analysis, Statistical Approach, Robust Fitting, Multiple Fitting, Image Grouping and Segmentation.

1 Introduction

In this paper, we propose a method for robustly recovering several instances of a curve model from a single noisy data set with severe perturbations (outliers). It is based on an extension of the work reported in [1], in which M-estimators are revisited in an Lagrangian formalism, leading to a new derivation and convergence proof of the well-known Iterative Reweighted Least Squares (IRLS) algorithm. Following the same approach based on the Lagrangian framework, we derive, in a natural way, a deterministic, alternate minimization algorithm for multiple regression, called Simultaneous Multiple Robust Fitting (SMRF) algorithm. The SMRF can be seen as an extension of the IRLS algorithm, in which an extra probability ratio, which is classical in clustering algorithms, appears in the expression of the weights. To tackle potential numerical issues, zero probabilities are banned in the computation of the weights and a Gaussian prior on the curves coefficients is introduced. Such a prior is, moreover, well-suited to sequential image processing and provides control on the curves. Applications to camera calibration and lane-markings tracking illustrate the effectiveness of the SMRF algorithm. In particular, it outperforms classical Gaussian mixture model algorithms in the presence of outliers.

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 121–133, 2008.

The paper is organized as follows. In Sec. 2, we present the robust multiple curves estimation problem and introduce our algorithmic strategy. The resulting algorithm is given in Sec. 3. Technical details on its derivation and convergence proof are given in the Appendix. In Sec. 4, connections are made with other approaches in the domain. Finally, we apply the algorithm to road tracking and to camera calibration, in Sec. 5.

2 Multiple Robust Maximum Likelihood Estimation (MLE)

In this section, we model the problem of simultaneously fitting m curves in a robust way. Each individual curve is explicitly described by a vector parameter $\tilde{A}_j$, $1 \leq j \leq m$. The observations, y, are given by a linear generative model:

$$y = X(x)^t \tilde{A}_j + b \tag{1}$$

where (x, y) are the image coordinates of a data point, $\tilde{A}_j = (\tilde{a}_{ij})_{0 \leq i \leq d}$ is the vector of curve parameters and $X(x) = (f_i(x))_{0 \leq i \leq d}$ is the vector of basis functions at the image coordinate x, which will be denoted as X for the sake of simplicity. These vectors are of size $d + 1$. Example of basis functions will be given in Sec. 5.2. Note that we consider the *fixed design* case, i.e. in (1), x is assumed non-random. In that case, it is shown that certain M-estimators attain the maximum possible breakdown point of approximately 50% [2]. In all that follows, the measurement noise b is assumed independent and identically distributed (iid) and centered. In order to render the estimates robust to non-Gaussian noise (outliers), we formulate the noise distribution as:

$$p_s(b) \propto \frac{1}{s} e^{-\frac{1}{2}\phi((\frac{b}{s})^2)} \tag{2}$$

where $\propto$ denotes the equality up to a factor, and s is the scale of the pdf. As stated in [3], the role of ϕ is to saturate the error in case of a large noise $|b| = |X^t \tilde{A}_j - y|$, and thus to lower the importance of outliers. The scale parameter, s, controls the distance from which noisy measurements have a good chance of being considered as outliers. The algorithm derivation is performed using the half-quadratic approach [4,5] revisited using classical optimization tools, namely Lagrange duality [1]. The potential function $\phi(t)$ must fulfill the following hypotheses:

- **H0**: defined and continuous on $[0, +\infty[$ as well as its first and second derivatives,
- **H1**: $\phi'(t) > 0$ (thus ϕ is increasing),
- **H2**: $\phi''(t) < 0$ (thus ϕ is concave).

Note that these assumptions are very different from those used in [3], where the convergence proof required that the potential function $\rho(b) = \phi(b^2)$ be convex. In the present case, the concavity and monotonicity requirements imply that $\phi'(t)$ is bounded, but $\phi(b^2)$ is not *necessarily convex* w.r.t. b.

Our goal is to simultaneously estimate the m curve parameter vectors $A_{j=1,\cdots,m}$ from the whole set of n data points (x_i, y_i), $i = 1, \cdots, n$. The probability of a measurement point (x_i, y_i), given the m curves is the sum of the probabilities over each curve:

$$p_i((x_i, y_i)|A_{j=1,\cdots,m}) \propto \frac{1}{s} \sum_{j=1}^{j=m} e^{-\frac{1}{2}\phi((\frac{X_i^t A_j - y_i}{s})^2)}.$$

Concatenating all curve parameters into a single vector $A = (A_j), j = 1, \cdots, m$ of size $m(d+1)$, we can write the probability of the whole set of points as the product of the individual probabilities:

$$p((x_i, y_i)_{i=1,\cdots,n}|A) \propto \frac{1}{s^n} \prod_{i=1}^{i=n} \sum_{j=1}^{j=m} e^{-\frac{1}{2}\phi((\frac{X_i^t A_j - y_i}{s})^2)} \tag{3}$$

Maximizing the likelihood $p((x_i, y_i)_{i=1,\cdots,n}|A)$ is equivalent to minimizing the negative of its logarithm:

$$e_{MLE}(A) = \sum_{i=1}^{i=n} -\ln(\sum_{j=1}^{j=m} e^{-\frac{1}{2}\phi((\frac{X_i^t A_j - y_i}{s})^2)}) + n\ln(s) \tag{4}$$

Using the same trick as the one described in [1] for robust fitting of a single curve, we introduce the auxiliary variables $w_{ij} = (\frac{X_i^t A_j - y_i}{s})^2$, as explained in the Appendix. We then rewrite the value $e_{MLE}(A)$ as the value achieved at the unique saddle point of the following Lagrange function:

$$L_R = \sum_{i=1}^{i=n} \sum_{j=1}^{j=m} \frac{1}{2}\lambda_{ij}(w_{ij} - (\frac{X_i^t A_j - y_i}{s})^2) + \sum_{i=1}^{i=n} \ln(\sum_{j=1}^{j=m} e^{-\frac{1}{2}\phi(w_{ij})}) - n\ln(s) \tag{5}$$

Then, the algorithm obtained by alternated minimizations of the dual function w.r.t. λ_{ij} and A is globally convergent, towards a local minimum of $e_{MLE}(A)$, as shown in the Appendix.

3 Simultaneous Multiple Robust Fitting Algorithm

As detailed in the Appendix, minimizing (5) leads to alternate between the three sets of equations:

$$w_{ij} = (\frac{X_i^t A_j - y_i}{s})^2, 1 \leq i \leq n, 1 \leq j \leq m, \tag{6}$$

$$\lambda_{ij} = \frac{e^{-\frac{1}{2}\phi(w_{ij})}}{\sum_{k=1}^{k=m} e^{-\frac{1}{2}\phi(w_{ik})}} \phi'(w_{ij}), 1 \leq i \leq n, 1 \leq j \leq m, \tag{7}$$

$$(\sum_{i=1}^{i=n} \lambda_{ij} X_i X_i^t) A_j = \sum_{i=1}^{i=n} \lambda_{ij} y_i X_i, 1 \leq j \leq m \tag{8}$$

In practice, some care must be taken, to avoid numerical problems and singularities. First, it is important that the denominator in (7) be numerically non-zero, which might occur for a data point located far from all curves. Zero probabilities are banned by adding a small value ϵ (equal to the machine precision) to the exponential in the probability p_i of a measurement point. As a consequence, when a point with index i is far from all curves, $\phi'(w_{ij})$ is weighted by a constant factor, $1/m$, in (7).

Second, the linear system in (8) can be singular. To avoid this, it is necessary to enforce a Gaussian prior on the whole curves parameters with bias A^{pr} and covariance matrix C^{pr}. Note that the reason for introducing such a prior is not purely technical: it is indeed a very simple and useful way of taking into account application-specific *a priori* knowledge, as shown in Sec. 5.3 and 5.4. As a default prior, we suppose that the bias is zero, i.e $A^{pr} = 0$, and that the inverse covariance matrix is block diagonal where each diagonal block equals:

$$C^{pr\,-1} = r \int_{-1}^{1} X(x) X(x)^t dx \tag{9}$$

assuming that $[-1, 1]$ is the range where x varies. The integral is the inverse covariance matrix of the curve fitting estimator under a Gaussian noise assumption which can be used for approximately modeling the truncation errors due to image sampling. The default prior also accounts for possible correlations between basis functions, which can be helpful when using non-orthogonal bases. The regularization term $(A - A^{pr})^t C^{pr\,-1} (A - A^{pr})$ is added to (4) and (5). Therefore, the parameter r controls the balance between the data fidelity term and the prior.

Finally, the Simultaneous Multiple Robust Fitting algorithm (SMRF) is:

1. Initialize the number of curves m, the vector $A^0 = (A_j^0)$, $1 \leq j \leq m$, which gathers all curves parameters and set the iteration index to $k = 1$.
2. For all indexes i, $1 \leq i \leq n$, and j, $1 \leq j \leq m$, compute the auxiliary variables $w_{ij}^k = (\frac{X_i^t A_j^{k-1} - y_i}{s})^2$ and the weights $\lambda_{ij}^k = \frac{\epsilon + e^{-\frac{1}{2}\phi(w_{ij}^k)}}{m\epsilon + \sum_{j=1}^{j=m} e^{-\frac{1}{2}\phi(w_{ij}^k)}} \phi'(w_{ij}^k)$.
3. Solve the linear system:

$$\left[D + C^{pr\,-1}\right] A^k = \begin{bmatrix} \sum_{i=1}^{i=n} \lambda_{i1}^k y_i X_i \\ \vdots \\ \sum_{i=1}^{i=n} \lambda_{im}^k y_i X_i \end{bmatrix} + C^{pr\,-1} A^{pr}.$$

4. If $\|A^k - A^{k-1}\| > \epsilon'$, increment k, and go to 2, else the solution is $A = A^k$.

In the above algorithm, D is the block-diagonal matrix whose m diagonal blocks are the matrices $\sum_{i=1}^{i=n} \lambda_{ij}^k X_i X_i^t$ of size $(d+1) \times (d+1)$, with $1 \leq j \leq m$. The prior covariance matrix C^{pr} is of size $m(d+1) \times m(d+1)$. The prior bias A^{pr} is a vector of size $m(d+1)$, as well as A and A^k. The complexity is $O(nm)$ for the step 2, and $O(m^2(d+1)^2)$ for the step 3 of the algorithm.

4 Connections with Other Approaches

The proposed algorithm has important connections with previous works in the field of regression and clustering and we would like to highlight a few of them.

In the single curve case, $m = 1$, the SMRF algorithm is reduced to the so-called Iterative Reweighted Least Squares extensively used in M-estimators [3], half-quadratic theory [5,4], and others. The SMRF and IRLS algorithms share very similar structures

and it is important to notice that the main difference lies within the Lagrange multipliers λ_{ij}, see (7). Compared to the IRLS, the λ_{ij} are just weighted by an extra probability ratio, which is classical in clustering algorithms.

To make the connection with clustering clearer, let us substitute $Y = A_j + b$ to the generative model (1), where Y and A_j are vectors of same size and respectively denote a data points and a cluster centroid. The derivation described in Sec. 3 is still valid and the obtained algorithm turns to be a clustering algorithm with m clusters, each cluster being represented by a vector, its centroid. The probability distribution of a cluster around its centroid is directly specified by the function ϕ. The obtained algorithm is thus able of modeling the Y_i's by a mixture of pdfs which are not necessarily Gaussian. The mixture problem is usually solved by the well-known Expectation-Minimization (EM) approach [6]. In the non-Gaussian case, the minimization (M) step implements robust estimation, which is an iterative process in itself. Hence, the resulting EM algorithm involves two nested loops, while the proposed algorithm features only one. An alternative to the EM approach is the Generalized EM (GEM) approach which consists in performing an approximate M-step: typically, only one iteration rather than the full minimization. The resulting algorithm in the robust case is identical to the one we derived within the Lagrangian framework (apart from the regularization of the singular cases). In our formalism however, no approximation is made in the derivation of the algorithm, in contrast with the GEM approach.

We also found that the SMRF algorithm is very close to an earlier work in the context of clustering [7]. However, to our knowledge, the latter was just introduced as an extra *ad-hoc* weighting over M-estimators without statistical interpretation and, moreover, singular configurations were not dealt with.

The SMRF algorithm is subject to the initialization problem since it only converges towards a local minimum. To tackle this problem, the Graduated Non Convexity approach (GNC) [8] is used to improve the chances of converging towards the global minimum. Details are given in Sec. 5.4. The SMRF can be also used as a fitting process within the RANSAC [9] approach to improve the convergence towards the global minimum.

5 Experimental Results

The proposed approach being based on a linear generative model, many applications could potentially be addressed using the SMRF algorithm. In this paper, we focus on two specific applications, namely simultaneous lane-markings tracking and camera calibration from a regular lattice of lines with geometric distortions. See [13] for more detail information.

5.1 Noise Model

Among the suitable functions for robust estimation, we use a simple parametric family of probability distribution functions, that was introduced in [1] under the name of *smooth exponential family* (SEF), $S_{\alpha,s}$:

$$S_{\alpha,s}(b) \propto \frac{1}{s} e^{-\frac{1}{2}\phi_\alpha((\frac{b}{s})^2)} \tag{10}$$

where, with $t = (\frac{b}{s})^2$, $\phi_\alpha(t) = \frac{1}{\alpha}((1+t)^\alpha - 1)$.

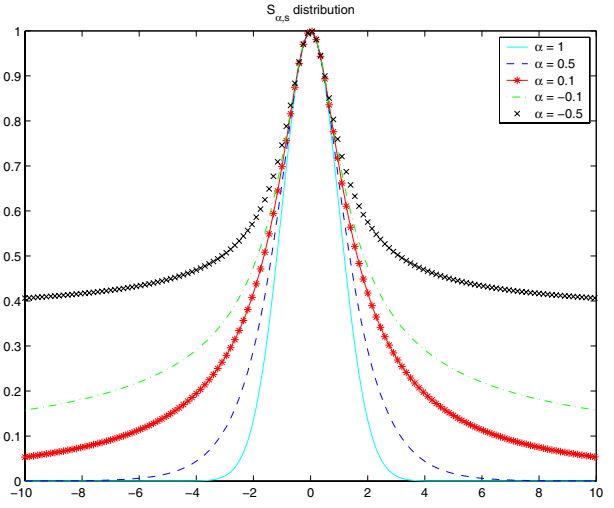


Fig. 1. Noise models in the SEF $S_{\alpha,s}$. Notice how the tails become heavier as α decreases.

These laws are shown in Figure 1 for different values of α. The smaller the value of α, the higher the probability of observing large, not to say very large, errors (outliers). This parameter allows a continuous transition between well-known statistical laws such as Gauss ($\alpha = 1$), smooth Laplace ($\alpha = \frac{1}{2}$) and T-Student ($\alpha \to 0$). This can be exploited to get better convergence of the SMRF algorithm by using the GNC approach, i.e. by progressively decreasing α towards 0.

5.2 Road Shape Model

The road shape features (x, y) are given by the lane-marking centers extracted using the local feature extractor described in [10]. An example of extraction is shown in Figure 6(b). In practice, we model road lane markings by polynomials $y = \sum_{i=0}^{d} a_i x^i$. Moreover, in the *flat world* approximation, the image of a polynomial on the road under perspective projection is a *hyperbolic polynomial* with equation $y = c_0 x + c_1 + \sum_{i=2}^{d} \frac{c_i}{(x - x_h)^i}$, where c_i is linearly related to a_i. Therefore, the hyperbolic polynomial model is well suited to the case of road scene analysis. To avoid numerical problems, a whitening of the data is performed by scaling the image in a $[-1, 1] \times [-1, 1]$ box for polynomial curves and in a $[0, 1] \times [-1, 1]$ box for hyperbolic polynomials, prior to the fitting.

5.3 Geometric Priors

As noticed in Sec. 3, the use of a Gaussian prior allows introducing useful application-specific knowledge. For example, using (9) for the diagonal blocks of the inverse prior covariance matrix, we take into account perturbations due to image sampling.

Tuning the diagonal elements of C^{pr} provides control on the curve degree. For polynomials, the diagonal components of the covariance matrix correspond to monomials of different degrees. The components of degree higher than one are thus set to smaller values than those of degree zero and one.

Geometric smooth constraints between curves can be enforced by using also non-zero off-diagonal blocks. In particular, it is a soft way of maintaining parallelism between curves. As an illustration, considering two lines $y = a_0 + a_1 x$ and $y = a'_0 + a'_1 x$, the prior covariance matrix is obtained by rewriting $(a_1 - a'_1)^2$ in matrix notations:

$$\begin{bmatrix} a_0 \\ a_1 \\ a'_0 \\ a'_1 \end{bmatrix}^t \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a'_0 \\ a'_1 \end{bmatrix}$$

The above matrix, multiplied by an overall factor can be used as an inverse prior covariance $C^{pr\,-1}$. The factor controls the balance between the data fidelity term and the other priors. Other kinds of geometric smooth constraints can be handled in a similar way, such as intersection at a given point, or symmetric orientations. These geometric priors can be combined by adding the associated regularization term $(A - A^{pr})^t C^{pr\,-1}(A - A^{pr})$ to (4) and (5).

5.4 Lane-Markings Tracking

We shall now describe the application of the SMRF algorithm to the problem of tracking lane markings.

In addition to the previous section, another interesting feature of using a Gaussian prior is that the SMRF is naturally suitable for being included in a Kalman filtering. However, this raises the question of the definition of the posterior covariance matrix of the estimate. Under the Gaussian noise assumption, the estimate of the posterior covariance matrix is well-known for each curve: $C_j = s^2 \left(\sum_{i=1}^{i=n} X_i X_i^t \right)^{-1}$. Unfortunately, in the context of robust estimation, the estimation of C_j for each curve A_j is a difficult issue and only approximate matrices are available. In [10], several approximates were compared. The underlying assumption for defining all these approximates is that the noise is independent. However, we found out that in practice, the noise is correlated from one image line to another. Therefore, all these approximates can be improved by introducing an had-hoc correction factor which accounts for data noise correlations in the inverse covariance matrix. We experimentally found that the following factor is appropriate, for each curve j:

$$1 - \frac{\sum_{i=1}^{i=n-1} \sqrt{\lambda_{ij} w_{ij} \lambda_{i+1,j} w_{i+1,j}}}{\sum_{i=1}^{i=n} \lambda_{ij} w_{ij}}$$

The approximate posterior covariance matrix for the whole set of curve parameters, A, is simply built as a block-diagonal matrix made of the individual posterior covariance matrices for each curve, C_j. This temporal prior can be easily combined with geometric priors for tracking parallel curves, for instance.

Fig. 2. Detected lane-markings (in green) and uncertainty about curve position (in red)

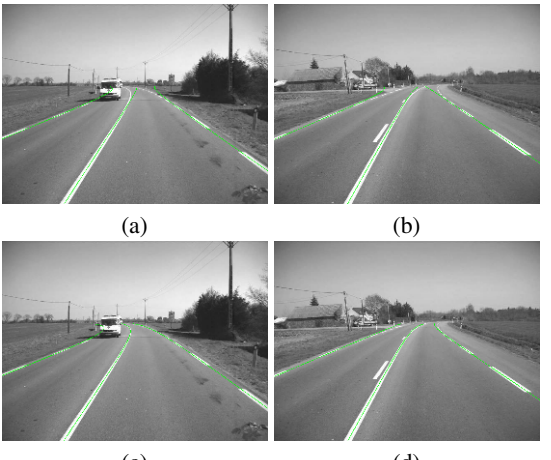


Fig. 3. Two images extracted from a sequence of 240 images processed with, on (a)(b), separate Kalman filters and, on (c)(d), simultaneous Kalman filter. The three detected lane-markings of degree two are in green.

Figure 2 shows the three curves simultaneously fitted on the lane-marking centers (in green) and the associated uncertainty curves of the horizontal position of each fitted curve ($\pm\sqrt{X(x)^t C_j^{-1} X(x)}$, in red). Notice that the uncertainty on the right sparse lane-marking is higher than for the continuous one on the center. Moreover, the higher the distance to the camera, the higher the uncertainty, since the curve gets closer to possible outliers. In all these experiments, and following, the parameters used for the noise model are $\alpha = 0.1$ and $s = 4$.

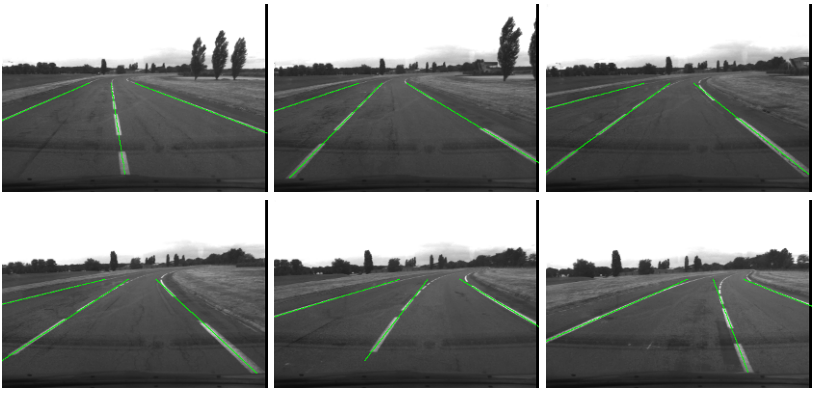


Fig. 4. Six of a 150-image sequence, featuring lane changes. Green lines show the three fitted lane-markings centers.

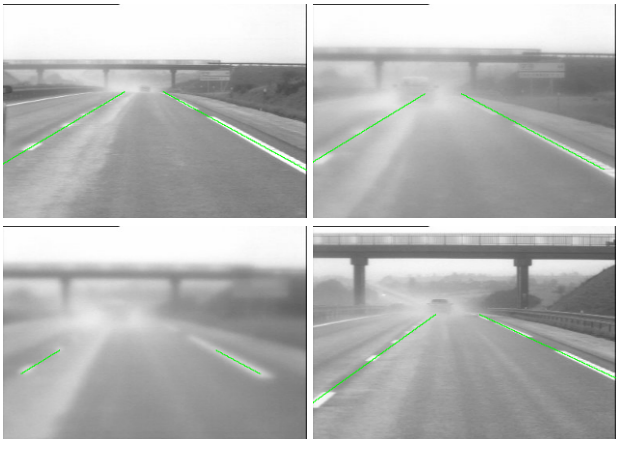


Fig. 5. Fitting in adverse conditions: in this excerpt, the left lane-marking is mostly hidden on two successive images

For the tracking itself, we experimented both separate Kalman filters on individual curves, and a simultaneous Kalman filter. The former can be seen as a particular case of the latter, in which the inverse prior covariance matrix C^{pr} is block-diagonal so the linear system of size $m(d+1)$ in the SMRF algorithm can be decomposed as m linear independent systems of size $d+1$. Figure 3 compares the results obtained with separate and simultaneous Kalman filters. Notice how the parallelism between curves is better preserved within the simultaneous Kalman filter, thanks to an adequate choice of the off-diagonal blocks of C^{pr}.

Figure 4 illustrates the ability of the SMRF-based Kalman filter to fit and track several curves in an image sequence. In that case, three lane-markings are simultaneously fitted and correctly tracked, even though the vehicle performs several lane changes

during the 150-image sequence. Notice that, while Kalman filtering can incorporate a dynamic model of the vehicle, we only used a static model in these experiments, since only the images were available. We observed that it is better to initialize the SMRF algorithm with the parameters resulting from the fitting on the previous image, rather than with the filtered parameters: filtering indeed introduces a delay in the case of fast displacements or variations of the tracked curves.

Moreover, we obtained interesting results on difficult road sequences. For instance, Figure 5 shows a short sequence of poor quality images, due to rain. The left lane-marking is mostly hidden on two consecutive images. Thanks to the simultaneous Kalman filter, the SMRF algorithm is able to interpolate correctly the hidden lane-marking.

5.5 Camera Calibration

We now present another application of the SMRF algorithm, in the context of camera calibration. The goal is to estimate accurately the position and orientation of the camera with respect to the road and its intrinsic parameters. A calibration setup made of two sets of perpendicular lines painted on the road is thus observed by a camera mounted on a vehicle, as shown in Figure 6(a). The SMRF algorithm can be used to provide accurate data to the calibration algorithm by estimating the grid intersections. Even though the markings are clearly visible in the image, some of them are quite short, and there are outliers due to the presence of water puddles. Figure 6(d) shows the extracted lane-marking centers. When a Gaussian mixture model is used, the obtained fit

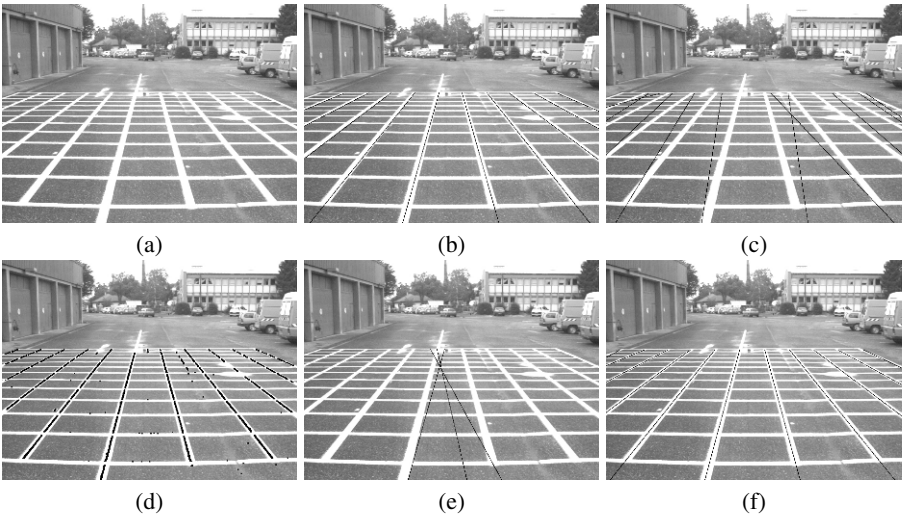


Fig. 6. (a) Original image of the calibration grid. (d) Extracted lane-marking centers (outliers are due to puddles). (b) 10 initial lines for the fitting on the vertical markings. (e) Fitted lines on the vertical markings under Gaussian noise assumption. (c) 12 initial lines for the fitting on the vertical markings. (f) The robust fitting yields 11 different correct lines.

is severely troubled by the outliers, as displayed in Figure 6(e), even though the curves are initialized very close to the expected solution, see Figure 6(b).

On the contrary, with the same extracted lane-marking centers, the SMRF algorithm, with noise model parameters $\alpha = 0.1$ and $s = 4$, leads to nice results, as shown in Figure 6(f) for the vertical lines. 11 different lines were obtained for the vertical markings, even if the initial curves where not very close to the solution as illustrated by Figure 6(c).

6 Conclusions

In the continuing quest for achieving robustness in detection and tracking curves in images, this paper makes two contributions. The first one is the derivation, in a MLE approach and using Kuhn and Tucker's classical theorem, of the so-called SMRF algorithm. This algorithm extends mixture model algorithm, such as the one derived using EM, to robust curve fitting. It is also an extended version of the IRLS, in which the weights incorporate an extra probability ratio. The second contribution is the regularization of the SMRF algorithm by introducing Gaussian priors on curve parameters and the handling of potential numerical issues by banning zero probabilities in the computation of weights. From our experiments, banning zero probabilities seems to have important positive consequences in pushing the curves to spread out all the data, and thus in providing improved robustness to the initialization, as shown in the context of camera calibration. The introduction of the Gaussian prior is also beneficial in particular in the context of image sequence processing, as illustrated with an application of simultaneous lane-markings tracking on-board a vehicle in adverse conditions. The approach being based on a linear generative model, it is quite generic and we believe that it can be used with benefits in many other fields of computer vision, such as clustering or appearance modeling, registration, parametric region fitting, as illustrated in [11,12,13].

References

1. Tarel, J.-P., Ieng, S.-S., Charbonnier, P.: Using robust estimation algorithms for tracking explicit curves. In: European Conference on Computer Vision (ECCV 2002), Copenhagen, Danmark, vol. 1, pp. 492–507 (2002)
2. Mizera, I., Müller, C.: Breakdown points and variation exponents of robust M-estimators in linear models. The Annals of Statistics 27(4), 1164–1177 (1999)
3. Huber, P.J.: Robust Statistics. John Wiley and Sons, New York (1981)
4. Geman, D., Reynolds, G.: Constrained restoration and the recovery of discontinuities. IEEE Transactions on Pattern Analysis and Machine Intelligence 14(3), 367–383 (1992)
5. Charbonnier, P., Blanc-Féraud, L., Aubert, G., Barlaud, M.: Deterministic edge-preserving regularization in computed imaging. IEEE Transactions on Image Processing 6(2), 298–311 (1997)
6. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, Series B (Methodological) 39(1), 1–38 (1977)
7. Cambell, N.A.: Mixture models and atypical values. Mathematical Geology 16(5), 465–477 (1984)

8. Blake, A., Zisserman, A.: Visual Reconstruction. MIT Press, Cambridge (1987)
9. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, Cambridge (2004); ISBN: 0521540518
10. Ieng, S.-S., Tarel, J.-P., Charbonnier, P.: Evaluation of robust fitting based detection. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3024, pp. 341–352. Springer, Heidelberg (2004)
11. Bigorgne, E., Tarel, J.-P.: Backward segmentation and region fitting for geometrical visibility range estimation. In: Yagi, Y., Kang, S.B., Kweon, I.S., Zha, H. (eds.) ACCV 2007, Part II. LNCS, vol. 4844, pp. 817–826. Springer, Heidelberg (2007)
12. Tarel, J.-P., Ieng, S.-S., Charbonnier, P.: Accurate and robust image alignment for road profile reconstruction. In: Proceedings of IEEE International Conference on Image Processing (ICIP 2007), San Antonio, Texas, USA, vol. V, pp. 365–368 (2007)
13. Tarel, J.-P., Ieng, S.-S., Charbonnier, P.: Robust Lane Marking Detection by the Half Quadratic Approach. Collection Etudes et Recherches des Laboratoires des Ponts et Chaussées, CR 49, LCPC (November 2007)
14. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
15. Minoux, M.: Mathematical Programming: Theory and Algorithms. John Wiley and Sons, Chichester (1986)
16. Luenberger, D.G.: Introduction to linear and nonlinear programming. Addison Wesley, Reading (1973)

Appendix

We shall first rewrite the value $-e_{MLE}(A)$ for any given $A = (A_j), j = 1, \cdots, m$ as the value achieved at the minimum of a convex problem under convex constraints. This is obtained by introducing the auxiliary variables $w_{ij} = (\frac{X_i^t A_j - y_i}{s})^2$. This apparent complication is in fact valuable since it allows introducing Lagrange multipliers, and thus to decompose the original problem in simpler problems. The value $-e_{MLE}(A)$ can be seen as the minimum value, w.r.t. $W = (w_{ij})_{1 \leq i \leq n, 1 \leq j \leq m}$, of:

$$E(A, W) = \sum_{i=1}^{i=n} ln(\sum_{j=1}^{j=m} e^{-\frac{1}{2}\phi(w_{ij})})$$

subject to nm constraints $h_{ij}(A, W) = w_{ij} - (\frac{X_i^t A_j - y_i}{s})^2 \leq 0$. This is proved by showing that the bound on each w_{ij} is always achieved. Indeed $E(A, W)$ is decreasing w.r.t. each w_{ij}, since its first derivative:

$$\frac{\partial E}{\partial w_{ij}} = -\frac{1}{2} \frac{e^{-\frac{1}{2}\phi(w_{ij})}}{\sum_{k=1}^{k=m} e^{-\frac{1}{2}\phi(w_{ik})}} \phi'(w_{ij})$$

is always negative, due to (H1).

To prove the local convergence of the SMRF algorithm in Sec. 3, we now focus on the minimization of $E(A, W)$ w.r.t. W only, subject to the nm constraints $h_{ij}(A, W) \leq 0$, w.r.t. W, for any A. We now introduce a classical result of convex analysis [14]: the function $g(Z) = log(\sum_{j=1}^{j=m} e^{z_j})$ is convex. Due to (H1) and (H2), $-\phi(w)$ is convex

and decreasing. Therefore, $E(A,W)$ w.r.t. W is convex as a sum of functions g composed with $-\phi$, see [14]. As a consequence, the minimization of $E(A,W)$ w.r.t. W is well-posed because it is a minimization of a convex function subject to convex (linear) constraints. We are thus allowed to apply Kuhn and Tucker's classical theorem [15]: if a solution exists, the minimization of $E(A,W)$ w.r.t. W is equivalent to searching from the unique saddle point of the Lagrange function of the problem:

$$L_R(A,W,\Lambda) = \sum_{i=1}^{i=n} ln(\sum_{j=1}^{j=m} e^{-\frac{1}{2}\phi(w_{ij})}) + \sum_{i=1}^{i=n}\sum_{j=1}^{m} \frac{1}{2}\lambda_{ij}(w_{ij} - (\frac{X_i^t A_j - y_i}{s})^2)$$

where $\Lambda = (\lambda_{ij}), 1 \leq i \leq n, 1 \leq j \leq m$ are Kuhn and Tucker multipliers ($\lambda_{ij} \geq 0$). More formally, we have proved for any A:

$$-e_{MLE}(A) = \min_W \max_\Lambda L_R(A,W,\Lambda) \tag{11}$$

Notice that the Lagrange function L_R is quadratic w.r.t. A, unlike the original error e_{MLE}. Using the saddle point property, we can change the order of variables W and Λ in (11). We now introduce the *dual function* $\mathcal{E}(A,\Lambda) = \min_W L_R(A,W,\Lambda)$, and rewrite the original problem as the equivalent following problem:

$$\min_A e_{MLE}(A) = \min_{A,\Lambda} -\mathcal{E}(A,\Lambda)$$

The algorithm consists in minimizing $-\mathcal{E}(A,\Lambda)$ w.r.t. A and Λ alternately. $\min_\Lambda -\mathcal{E}(A,\Lambda)$ leads to Kuhn and Tucker's conditions:

$$\lambda_{ij} = \frac{e^{-\frac{1}{2}\phi(w_{ij})}}{\sum_{k=1}^{k=m} e^{-\frac{1}{2}\phi(w_{ik})}}\phi'(w_{ij}) \tag{12}$$

$$w_{ij} = (\frac{X_i^t A_j - y_i}{s})^2 \tag{13}$$

and $\min_{A_j} -\mathcal{E}(A,\Lambda)$ leads to:

$$(\sum_{i=1}^{i=n} \lambda_{ij} X_i X_i^t) A_j = \sum_{i=1}^{i=n} \lambda_{ij} y_i X_i,\ 1 \leq j \leq m \tag{14}$$

Using classical results, see e.g. [15], $-\mathcal{E}(A,\Lambda)$ is proved to be convex w.r.t. Λ. The dual function is clearly quadratic and convex w.r.t. A. As a consequence, this implies that such an algorithm always strictly decreases the dual function if the current point is not a stationary point (i.e a point where the first derivatives are all zero) of the dual function [16]. The problem of stationary points is easy to solve by checking the positiveness of the Hessian matrix of $\mathcal{E}(A,\Lambda)$. If this matrix is not positive, we disturb the solution so that it converges to a local minimum. This proves that the algorithm is globally convergent, i.e, it converges toward a local minimum of $e_{MLE}(A)$ for all initial A_0's which are neither a maximum nor a saddle point.

Part III
Image Understanding

A Dempster-Shafer Theory Based Combination of Classifiers for Hand Gesture Recognition

Thomas Burger[1,3], Oya Aran[2], Alexandra Urankar[3], Alice Caplier[1], and Lale Akarun[2]

[1] Gipsa-Lab, Institut National Polytechnique de Grenoble, 46 av. Felix Viallet, Grenoble, France
{firstname.lastname}@gipsa-lab.inpg.fr
[2] Dep. of Comp. Eng., Bogazici University, Bebek 34342, Istanbul, Turkey
aranoya@boun.edu.tr, akarun@boun.edu.tr
[3] France Telecom R&D employee during this research conduction, France

Abstract. As part of our work on hand gesture interpretation, we present our results on hand shape recognition. Our method is based on attribute extraction and multiple partial classifications. The novelty lies in the fashion the fusion of all the partial classification results are performed. This fusion is (1) more efficient in terms of information theory and leads to more accurate results, (2) general enough to allow heterogeneous sources of information to be taken into account: Each classifier output is transformed to a belief function, and all the corresponding functions are fused together with other external evidential sources of information.

Keywords: SVM, Expert systems, HMM, Belief functions, Hu invariants, Hand shape and gesture recognition, Cued Speech, probabilistic transform.

1 Introduction

Hand shape recognition is a widely studied topic which has a wide range of applications such as HCI [1], automatic gesture translators, tutoring tools for the hearing-impaired [2], [3], augmented reality, and medical image processing.

Even if this field is dominated by Bayesian methods, several recent studies deal with evidential methods, as they bring a certain advantage in the fashion uncertainty is processed in the decision making [4], [5].

The complete recognition of a hand shape with no constraint on the shape is an open issue. Hence, we focus on the following problem: (1) the hand is supposed to roughly remain in a plan which is parallel to the acquisition plan (2) only nine different shapes are taken into account (**Fig. 1**a). No assumptions are made on the respective location of the fingers (whether they are gathered or not, which drastically increases the inner variance of each shape); except for hand shapes 2 (gathered fingers) and 8 (as separated as possible), as this is their only difference. These nine hand shapes correspond to the gesture set used in Cued Speech [6].

There are many methods already developed to deal with hand modeling and analysis. For a complete review, see [7], [8]. In this paper, we do not develop the segmentation

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 137–150, 2008.

(a) Artificial representation of the hand shape classes.

(b) Segmented hand shape examples from real data.

Fig. 1. The 9 hand shape classes

aspect. Hence, we consider several corpora of binary images such as in Fig. 1b, as the basis of our work. The attribute extraction is presented in Section 2. The required classification tools are presented in Section 3. Section 4 is the core of this paper: we apply the decision making method, which is theoretically presented in [9], to our specific problem, and we use its formalism as a framework in which it is possible to combine classifiers of various nature (SVMs and expert systems) providing various partial information. Finally, Section 5 provides experimental results, and Section 6 discusses possible theoretical extensions.

2 Attribute Definition

2.1 Hu Invariants

The dimensionality of the definition space for the binary images we consider is very large, and it is intractable to use pixel values to perform the classification. One needs to find a more compact representation of the image. Several such binary image descriptors are to be found in the image compression literature [10]. They can be classified into two main categories:

- *Region descriptors*, which describe the binary mask of a shape, such as Zernike moments, Hu invariants, and grid descriptors.
- *Edge descriptors*, which describe the closed contour of the shape, such as Fourier descriptors, and Curvature Scale Space (CSS) descriptors.

Region descriptors are less sensitive to edge noise because of an inertial effect of the region description. Edge descriptors are more related to the way human compare shapes.

A good descriptor is supposed to obey several criteria, such as geometrical invariance, compactness, being hierarchical (so that the precision of the description can be truncated), and being representative of the shape.

We focus on Hu invariants, which are successful in representing hand shapes [11]. Note that parallel studies are conducted on promising Fourier-Mellin Descriptors. The Hu invariants' purpose is to express the mass repartition of the shape via several

inertial moments of various orders, on which specific transforms ensure invariance to similarities.

Let us compute the classical definition of centered inertial moments of order $p+q$, for the shape (invariant to translation, as they are centered on the gravity center):

$$m_{pq} = \iint_{x\,y} (x-\overline{x})^p \, (y-\overline{y})^q \, \delta(x,y)\, dx\, dy \tag{1}$$

with $\overline{x}$ and $\overline{y}$ being the coordinates of the center of gravity of the shape and $\delta(x,y)=1$ if the pixel belongs to the hand shape and 0 otherwise. In order to make these moments invariant to scale, we normalize them:

$$n_{pq} = \frac{m_{pq}}{m_{00}^{\frac{p+q}{2}+1}} \tag{2}$$

Then, we compute the six Hu invariants, which are invariant to rotation, and mirror reflection:

$$\begin{aligned}
S_1 &= n_{20} + n_{02} \\
S_2 &= (n_{20} + n_{02})^2 + 4 \cdot n_{11}^{\,2} \\
S_3 &= (n_{30} - 3 \cdot n_{12})^2 + (n_{03} - 3 \cdot n_{21})^2 \\
S_4 &= (n_{30} + n_{12})^2 + (n_{03} + n_{21})^2 \\
S_5 &= (n_{30} - 3 \cdot n_{12}) \cdot (n_{30} + n_{12}) \cdot \left((n_{30} + n_{12})^2 - 3 \cdot (n_{03} + n_{21})^2\right) \\
&\quad - (n_{03} - 3 \cdot n_{21}) \cdot (n_{03} + n_{21}) \cdot \left(3 \cdot (n_{30} + n_{12})^2 - (n_{03} + n_{21})^2\right) \\
S_6 &= (n_{20} + n_{02}) \cdot \left((n_{30} + n_{12})^2 - (n_{03} + n_{21})^2\right) \\
&\quad + 4 \cdot n_{11}^{\,2} \cdot (n_{30} + n_{12}) \cdot (n_{03} + n_{21})
\end{aligned} \tag{3}$$

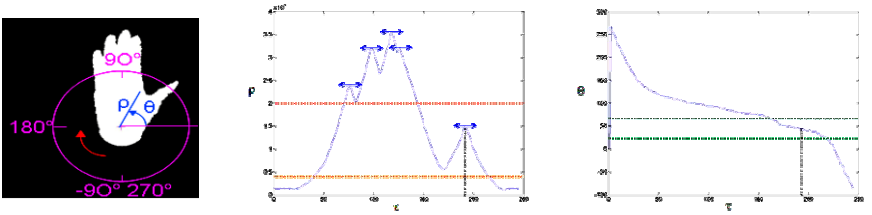
A seventh invariant is available. Its sign permits to discriminate mirror images and thus, to suppress the reflection invariance:

$$\begin{aligned}
S_7 &= (3 \cdot n_{21} - n_{03}) \cdot (n_{30} + n_{12}) \cdot \left((n_{30} + n_{12})^2 - 3 \cdot (n_{03} + n_{21})^2\right) \\
&\quad - (n_{30} - 3 \cdot n_{12}) \cdot (n_{03} + n_{21}) \cdot \left(3 \cdot (n_{30} + n_{12})^2 + (n_{03} + n_{21})^2\right)
\end{aligned} \tag{4}$$

The reflection invariance has been removed at the acquisition level and only left hands are processed. Hence, we do not need to discriminate mirror images. We nevertheless use S_7 as both the sign and the magnitude carry information: it sometimes occurs that hand shapes 3 and 7 (both with separated fingers) really look like mirror images. Finally, the attributes are: $\{S_1, S_2, S_3, S_4, S_5, S_6, S_7\}$.

2.2 Thumb Presence

The thumb is an easy part to detect, due to its peculiar size and position with respect to the hand. Moreover, the thumb presence is a very discriminative piece of evidence as three hand shapes require the thumb and six do not require it. The thumb detector works as follows:



(a) Binary hand shape image.

(b) Hand polar parametric representation along the curvilinear abscissa τ (distance ρ and angle θ). The thumb is within a peculiar distance and angle range (horizontal lines).

Fig. 2. Thumb detection

(1) Polar parametric definition: By following the contour of the hand shape, a parametric representation $\{\rho(\tau), \theta(\tau)\}$ is derived in polar coordinates (Fig.2a)

(2) Peak detection: After smoothing the parametric functions, (low-pass filtering and sub-sampling), the local maxima of the ρ function are detected. Obviously, they correspond to the potential fingers (Fig.2b).

(3) Threshold adaptation: Thresholds must be defined on the distance and the angle values to indicate the region in which a thumb is plausible. The angle thresholds that describe this region are derived from morphological statistics [12] : in practice, the thumb angle with respect to the horizontal axis (Fig.2b) is between 20° and 65°. The distance thresholds are derived from a basic training phase whose main purpose is to adapt the default *approximate* values (1/9 and 5/9 of the hand length) via a scale normalization operation with respect to the length of the thumb. Even if the operation is simple, it is mandatory to do so, as the ratio of the thumb length with respect to the total hand length varies from hand to hand.

(4) Peak measurement: If a finger is detected in the area defined by these thresholds, it is the thumb. Its height with respect to the previous local minima (Fig.2) is measured. It corresponds to the height between the top of the thumb and the bottom of the inter-space between the thumb and the index. This value is the *thumb presence indicator* (it is set to zero when no thumb is detected). In practice, the accuracy of the thumb detection (the thumb is detected when the corresponding indicator has a non-zero value) reaches 94% of true detection with 2% of false alarms.

The seven Hu invariants and the thumb presence indicator are used as attributes for the classification.

3 Classification Tools

3.1 Belief Functions and the Transfer Belief Model (TBM)

In this section, we briefly present the necessary background on belief functions. For deeper comprehension of these theories, see [13] and [14].

Let Ω be the set of N exclusive hypotheses $h_1 \ldots h_N$. We call Ω the frame of discernment, or the frame, for short. Let $m(.)$ be a belief function on 2^Ω (the powerset of Ω) that represents our mass of belief in the propositions that correspond to the elements of 2^Ω:

$$\begin{aligned} m &: 2^{\Omega} \to [0,1] \\ A &\mapsto m(A) \quad \text{with} \sum_{A \subseteq \Omega} m(A) = 1 \end{aligned} \tag{5}$$

Note that:

−Belief can be assigned to non-singleton propositions, which allows modeling the hesitation between elements;

−In the TBM, it is possible to associate a belief in Ø. It corresponds to conflict in the model, throughout an assumption in an undefined hypothesis of the frame or throughout a contradiction between the information on which the decision is made.

To combine several belief functions (each associated to one specific captor) into a global belief function (under associativity and symmetry assumptions), one uses the conjunctive combination. For N belief functions, $m_1 \ldots m_N$, defined on the same frame Ω, the conjunctive combination is defined as:

$$\begin{aligned} (\cap) &: \overbrace{\mathfrak{B}^{\Omega} \times \mathfrak{B}^{\Omega} \times \ldots \times \mathfrak{B}^{\Omega}}^{N} \to \mathfrak{B}^{\Omega_v} \\ & m_1 \,(\cap)\, m_2 \,(\cap) \ldots (\cap)\, m_N \mapsto m_{(\cap)} \end{aligned} \tag{6}$$

with $\mathfrak{B}^{\Omega}$ being the set of belief functions defined on Ω and $m_{(\cap)}$ being the global combined belief function. Thus, $m_{(\cap)}$ is calculated as:

$$m_{(\cap)}(A) = \sum_{A = A_1 \cap \ldots \cap A_N} \left(\prod_{n=1}^{N} m_n(A_n) \right) \quad \forall A \subseteq 2^{\Omega_v} \tag{7}$$

The conjunctive combination means that, for each element of the power set, its belief is the combination of all the beliefs (from the N sources) which imply it: it is the evidential generalization of the logical AND.

After having fused several beliefs, the knowledge in the problem is modeled via a function over 2^{Ω}, which expresses the potential hesitations in the choice of the solution. In order to provide a complete decision, one needs to eliminate this hesitation. For that purpose, we use the *Pignistic Transform* [14], which maps a belief function from 2^{Ω} onto Ω, on which a decision is easy to make. The *Pignistic Transform* is defined as:

$$\mathrm{BetP}(h) = \frac{1}{1 - m(\emptyset)} \sum_{h \in A,\, A \subset \Omega} \frac{m(A)}{|A|} \quad \forall h \in \Omega \tag{8}$$

where A is a subset of Ω, or equivalently, an element of 2^{Ω}, and $|A|$ its cardinal when considered as a subset of Ω. This transform corresponds to sharing the hesitation between the implied hypotheses, and normalizing the whole by the conflictive mass.

As an illustration of all these concepts, let us consider a simple example: Assume that we want to automatically determine the color of an object. The color of the object can be one of the primary colors: red (R), green (G) or blue (B). The object is analyzed by two sensors of different kind, each giving an assumption of its color.

Table 1. Numerical example for belief function use

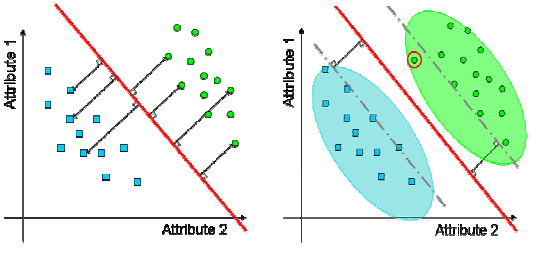
	∅	R	G	B	{R, G}	{R, B}	{B, G}	{R, G, B}
m_1	0	0.5	0	0	0.5	0	0	0
m_2	0	0	0	0	0	0	0.4	0.6
$m_1 (\cap) m_2$	0.2	0.3	0.2	0	0.3	0	0	0
BetP	0	**0.56**	0.44	0	0	0	0	0

The observations of the sensors are expressed as belief functions $m_1(.)$ and $m_2(.)$ and the frame is defined as Ω_{color} = {∅, R, G, B, {R, G}, {R, B}, {B, G}, {R, G, B}} representing the hypothesis about the color of the object. Then, they are fused together into a new belief function via the conjunctive combination. As the object has a single color, the belief in union of colors is meaningless from a decision making point of view. Hence, one applies the Pignistic Transform on which a simple *argmax* decision is made. This is summarized and illustrated in Table 1.

3.2 Support Vector Machines

SVMs [15], [16] are powerful tools for binary classification. Their purpose is to extract the correlation of the attributes for each class by defining a separating hyperplane derived from a training corpus, which is supposed to be statistically representative of the classes involved. The hyperplane is chosen among all the possible hyperplanes through a combinatorial problem optimization, so that it maximizes the distance (called the margin) between each class and the hyperplane itself (Fig. 3a & Fig. 3b).

To deal with the nine hand shapes in our database, a multiclass classification with SVMs must be performed. As SVMs are restricted to binary classification, several strategies are developed to adapt them for multiclass classification problems [17]. For



(a) (b)

Fig. 3. (a) Combinatorial optimization of the hyperplane position under the constraints of the training corpus item positions. (b) The SVM provides good classification despite the bias of the training.

that purpose, we have developed our own scheme [9], the Evidential Combination, which fuses the outputs of the SVMs using the belief formalism, and which provides a robust way of dealing with uncertainties. The method can be summarized by the following three steps:

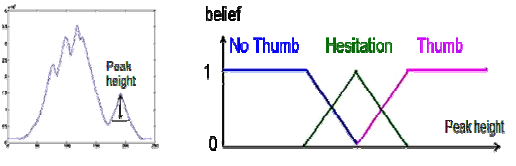
(1) 36 SVMs are used to compare all the possible class pairs among nine classes;
(2) A *belief function* is associated to each SVM output, to model the partial knowledge brought by the corresponding partial binary classification;
(3) The belief functions are fused together with a conjunctive combination, in order to model the complete knowledge of the problem, and to make a decision according to its value.

Classically, SVM outputs are +1 or -1, depending on the class chosen, but it remains a binary decision with respect of the two classes involved. Then, all the binary outputs are fused by a voting process. Unfortunetaly, (1) the votes are in ties for two or more classes, (2) the various outputs are not consistent: SVM_1 chooses class_A rather than class_B, SVM_2 chooses class_B rather than class_C, and SVM_3, class_C rather than class_A. In order to deal with such situations, methods have been proposed to convert the SVM outputs into probabilistic distributions, but, in [17], these methods are said to be equivalent to voting. On the contrary, our Evidential Combination has proved to be efficient on various datasets [9].

4 Decision Scheme

4.1 Belief in the Thumb Presence

In order to fuse the information from the thumb presence indicator with the output of the SVM classifier, one needs to represent it with a belief function. As it is impossible to have a complete binary certitude on the presence of the thumb (it is possible to be misled at the thumb detection stage as explained previously), we use a belief function which authorizes hesitation in some cases.



(a) (b)

Fig. 4. (a) The peak height determines (b) the belief in the presence of the thumb

From an implementation point of view, we use a technique based on fuzzy sets, as explained in Fig. 4: The higher the peak of the thumb is, the more confident we are in the thumb presence. This process is fully supported by the theoretical framework of belief functions, as the set of the finite fuzzy sets defined on Ω is a subset of 2^{Ω} (the set of belief functions defined on Ω). Moreover, as belief functions, fuzzy sets have

special properties which make them convenient to fuse with the conjunctive combination [18]. Note that it is roughly the same technique as the one used to associate a belief function to the output of each SVM.

The three values that define the support of the hesitation in **Fig. 4**b have been manually fitted according to observations on the training set. Making use of the "fuzziness" of the threshold between the thumb presence and absence, these values are not necessarily precisely settled. In practice, they are defined via three ratios (1/5, 1/20 and 1/100) of the distance between the center of palm and the furthest element from it of the contour.

Then, the belief in the presence of the thumb can be associated to a belief in some hand shapes to produce a partial classification: In hand shapes 0, 1, 2, 3, 4, and 8, there is no thumb, whereas it is visible for shapes 5, 6 and 7. In case of hesitation in the thumb presence, no information is brought and the belief is associated to Ω.

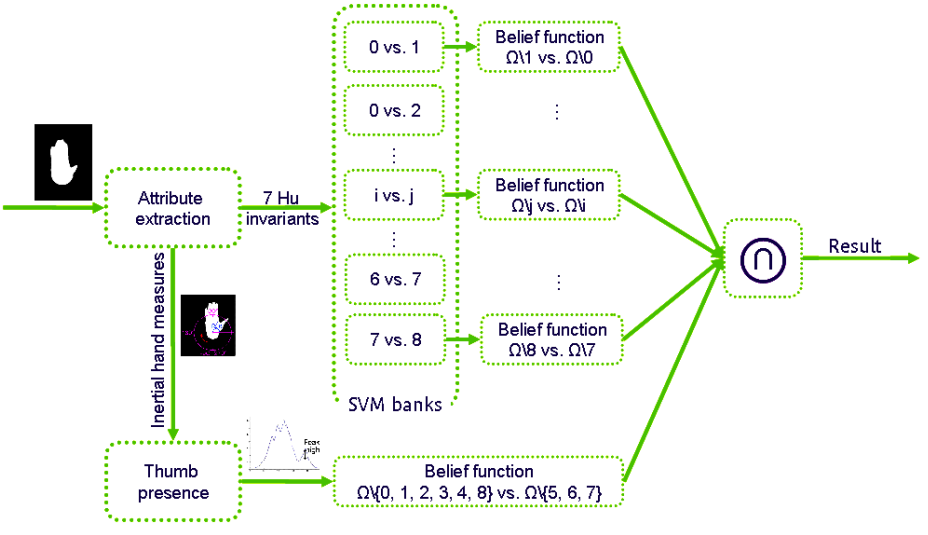


Fig. 5. Global fusion scheme for hand shape classification

4.2 Partial Classification Fusion

Thanks to the evidential formalism, it is possible to fuse partial information from various classifiers (36 SVMs and 1 expert system) through the conjunctive combination (Fig. 5). In that fashion, it is possible to consider a SVM-based system and integrate it into a wider data fusion system.

This fusion provides a belief function over the powerset 2^{Ω} of all the possible hand shapes Ω. This belief is mapped over Ω via the Pignistic Transform, to express our belief in each singleton element of Ω. Then, the decision is made by an *argmax* function over Ω.

$$D^{*} = \underset{\Omega}{\operatorname{argmax}}\left(\operatorname{BetP}(.)\right) \tag{9}$$

5 Results

In this section, we present various results on the methodology described above. In 5.1, the database and the evaluation methods are detailed. In 5.2, the experiments and corresponding results are given.

5.1 Database and Methodology

The hand shape database used in this work is obtained from Cued Speech videos. The transition shapes are eliminated manually and the remaining shapes are labeled and used in the database as binary images representing the 9 hand shapes (Fig. 1).

Table 2. Details of the database

Hand Shape	*Corpus 1 (Training set)*	*Corpus 2 (Test set)*
0	37	12
1	94	47
2	64	27
3	84	36
4	72	34
5	193	59
6	80	46
7	20	7
8	35	23
Total	679	291

The training and test sets of the database are formed such that there is no strict correlation between them. To ensure this, two different corpuses are used in which a single user is performing two completely different sentences using Cued Speech. The respective distribution of the two corpora are given in Table 2. The statistical distribution of the hand shapes is not balanced at all within each corpus. The reason of such a distribution is related to the linguistics of Cued Speech, and is beyond our scope.

For all the experiments, Corpus 1 is used as the training set for the SVMs and Corpus 2 is used as the test set. Since the real labels are known, we use the classical definition of the accuracy to evaluate the performance of the classifier:

$$Accuracy = 100 \cdot \frac{\text{Number Of Well Classified Items}}{\text{Total Number Of Items}} \tag{10}$$

To fairly quantify the performances of each classification procedure, two indicators are used: (1) The difference between the respective accuracies, expressed in the number of point $\Delta Point$, and (2) the percentage of avoided mistake $\%AvMis$:

$$\Delta Point = Accuracy(Method\_2) - Accuracy(Method\_1)$$

$$\begin{aligned} \%AvMis &= 100 \cdot \frac{\text{Number of Avoided Mistakes}}{\text{Total Number of Mistakes}} \\ &= 100 \cdot \frac{\Delta Point}{100 - Accuracy(Method\_1)} \end{aligned} \tag{11}$$

5.2 Experiments

The goal of the first experiment is to evaluate the advantage of the evidential fusion for the SVM. Thus, we compare the classical methods for SVM multi-classification to the one of [9]. For both of the methods, the training is the same and the SVMs are tuned with respect to the training set and the thumb information is not considered.

For the implementation of the SVM functionalities, we use the open source C++ library LIBSVM [19]. We use:

- C-SVM, which is an algorithm to solve the combinatorial optimization. The cost parameter is set to 100,000 and termination criteria to 0.001.
- Sigmoid kernels in order to transform the attribute space so that it is linearly separable:

$$\begin{aligned} &Ker_{\gamma,R}(u,v) = \tanh\left(\gamma \cdot u^T \cdot v + R\right) \\ &\text{with} \quad \gamma = 0.001 \quad \text{and} \quad R = -0.25 \end{aligned} \tag{12}$$

For the evidential method, we have made various modifications on the software so that the SVM output is automatically presented throughout the evidential formalism [9].

The results are presented in Table 3, as the test accuracy of the classical voting procedure and the default tuning of the evidential method. The improvement in $\Delta Point$ is worth 1.03 points and corresponds to an avoidance of mistakes of $\%AvMis = 11.11\%$.

Table 3. Results for experiments 1 & 2

	Classical Voting procedure	*Evidential method*	
		Default (no thumb detection)	With Thumb Detection
Test Accuracy	90.7%	91.8%	92.8%

The goal of the second experiment is to evaluate the advantage of the thumb information. For that purpose, we add the thumb information to the evidential method. Thus, the training set is used to set the two thresholds, which defines the possible distance with respect to the center of palm. However, the thumb information is not used during the training of the SVMs as they only work on the Hu invariants, as explained in Fig. 5. The results with and without the thumb indicator are presented in Table 3.

Table 4. Confusion matrix for the second method on Corpus 2, with the Thumb and NoThumb superclasses framed together

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
0	**12**	0	0	0	0	0	0	0	0
1	0	**46**	0	0	0	0	0	0	1
2	0	**2**	**23**	2	0	0	0	0	0
3	0	2	0	**29**	2	2	1	0	0
4	0	0	0	1	**32**	0	0	0	1
5	0	0	0	0	0	**58**	0	1	0
6	0	0	2	0	0	0	**41**	3	0
7	0	0	0	0	0	0	1	**6**	0
8	0	0	0	0	0	0	0	0	**23**

The evidential method that uses the thumb information provides an improvement of 2.06 points with respect to the classical voting procedure, which corresponds to an avoidance of 22.22% of the mistakes. Table 4 presents the corresponding confusion matrix for the test set: Hand shape 3 is often misclassified into other hand shapes, whereas, on the contrary, hand shape 1 and 7 gather a bit more misclassification from other hand shapes. Moreover, there are only three mistakes between THUMB and NO_THUMB super-classes.

6 Discussion and Theoretical Outlook

We have presented a method and experimental data which demonstrates that the Evidential Combination of SVM is an interesting tool to fuse SVM-processed data in a wider data fusion scheme. Consequently, the next challenge is to find a method so that other non-evidential classifier outputs can be considered in the evidential framework. In fact, it is required to guarantee that this method is available on SVM and expert systems, but also on other classifiers, in what we have called a "wider fusion scheme".

Of course, as explained in [9], the method also fits any binary classifier in which the minimum distance to the separating hyperplane is known (the margin), as it is the only required point to define the belief function. In case no such margins are available, it is possible to experimentally define them, via training or cross validation. In [20], we have applied this paradigm to video recognition of American Sign Language. Each sign is modeled by a Hidden Markov Model (HMM) and binary classifiers are derived from pairwise comparisons of the HMM likelihood scores. The very satisfying results show the efficiency of the method. But, it also demonstrates that it is not limited to binary classifiers; as a matter of fact, the derivation of binary classifiers from pairwise comparison is somehow artificial, and as a consequence, it is more straightforward to directly handle such unary classifiers (such as HMM). Hence, our method is also efficient on unary classifiers based on their generative properties.

Moreover, let us point out another fact: Practically, the result of a set of HMMs is a set of likelihood scores, i.e. a subjective probability distribution prior to any normalization.

Thus, this work is a good source of inspiration to provide a method to convert a probability into a belief function. As several such transformation methods are defined in incompatible fashions in the literature, such as [21], and as the relationships between belief functions and probabilities are a topic of strong debates [22], this subject should be carefully considered. Thus, we do not address it with respect to its epistemological dimension, and we stick to strict and simple computational considerations. As an introduction to further theoretical development, we propose to consider the following computation as an eventual interesting way to convert a probability distribution into a belief function: First of all, we make the assumption that the Pignistic Transform defines the relationships between probabilities and BF (which is a strong assumption on which the entire community does not agree). Moreover, we consider in this computational outlook that, any difference in the probability of two possible outcome gives an evidential clues that the outcome of higher probability is more likely to be believed in; and the quantification of the corresponding belief function is proportional to the difference of the probabilities (this later assumption is slightly more likely to be taken for granted). It is possible to derive *N*-1 belief functions from the comparison of *N* ordered probability (from the highest value to the smallest) values corresponding to *N* outcomes. The Pignistic Transform of their conjunctive combination should give the original probability distribution:

$$\mathrm{PigT}(m_1 \textcircled{\cap} m_2 \textcircled{\cap} \ldots \textcircled{\cap} m_{N\text{-}1}) = \mathrm{p}$$

with PigT corresponding to the pignistic transform. As, any of the m_i belief functions is defined on the powerset of two outcomes, it is possible to simplify the computation of the conjunctive combination (where $m_{(\cap)}$ corresponds to the results of the combination of the m_i) :

$$\begin{cases} m_{(\cap)}\left(\mathcal{C}^1\right) = m_1\left(\mathcal{C}^1\right) \\ m_{(\cap)}\left(\left\{\mathcal{C}^1, \mathcal{C}^2\right\}\right) = m_2\left(\left\{\mathcal{C}^1, \mathcal{C}^2\right\}\right) \cdot m_1(\Omega) \\ \vdots \\ m_{(\cap)}\left(\left\{\mathcal{C}^1, \ldots, \mathcal{C}^i\right\}\right) = m_i\left(\left\{\mathcal{C}^1, \ldots, \mathcal{C}^i\right\}\right) \cdot \prod_{j=1}^{i-1} m_j(\Omega) \\ \vdots \\ m_{(\cap)}(\Omega) = \prod_{j=1}^{N-1} m_j(\Omega) \\ m_{(\cap)}(A) = 0 \quad \forall \text{ other } A \in 2^{\Omega} \end{cases}$$

Then, the application of the pignistic transform gives:

$$\begin{cases} \mathrm{BetP}\left(\mathcal{C}^1\right) = m_{(\cap)}\left(\mathcal{C}^1\right) + \dfrac{m_{(\cap)}\left(\left\{\mathcal{C}^1, \mathcal{C}^2\right\}\right)}{2} + \cdots + \dfrac{m_{(\cap)}\left(\left\{\mathcal{C}^1, \ldots, \mathcal{C}^i\right\}\right)}{i} + \cdots + \dfrac{m_{(\cap)}(\Omega)}{N} \\ \mathrm{BetP}\left(\mathcal{C}^2\right) = \dfrac{m_{(\cap)}\left(\left\{\mathcal{C}^1, \mathcal{C}^2\right\}\right)}{2} + \cdots + \dfrac{m_{(\cap)}\left(\left\{\mathcal{C}^1, \ldots, \mathcal{C}^i\right\}\right)}{i} + \cdots + \dfrac{m_{(\cap)}(\Omega)}{N} \\ \vdots \\ \mathrm{BetP}\left(\mathcal{C}^i\right) = \dfrac{m_{(\cap)}\left(\left\{\mathcal{C}^1, \ldots, \mathcal{C}^i\right\}\right)}{i} + \cdots + \dfrac{m_{(\cap)}(\Omega)}{N} \\ \vdots \\ \mathrm{BetP}\left(\mathcal{C}^N\right) = \dfrac{m_{(\cap)}(\Omega)}{N} \end{cases}$$

where C^i corresponds to the i^{th} outcome. As we have assumed that $\mathrm{BetP}(C^i) = \mathrm{P}(C^i)$, one has :

$$m_i(\Omega) = 1 - i \cdot \frac{\mathrm{P}(C^i) - \mathrm{P}(C^{i+1})}{\prod_{k=1}^{i-1} m_k(\Omega)}$$

which can be iteratively computed, and which set the N-1 m_i with respect to the N-1 pairwise subtractions $\mathrm{P}(C^i)$–$\mathrm{P}(C^{i+1})$ of the probability distribution.

7 Conclusions

In this paper, we propose to apply a belief-based method for SVM fusion to hand shape recognition. Moreover, we integrate it in a wider classification scheme which allows taking into account other sources of information, by expressing them in the Belief Theories formalism. The results are better than with the classical methods (more than 1/5 of the mistakes are avoided) and the absolute accuracy is high with respect to the number of classes involved. Directions for future work are presented in the last section, where we derive from a computational point of view, a method that could potentially convert a probability distribution into a belief function. The consequences of this computation from an information theory point of view are not explored yet, so that it is not possible to assess the interest of this work yet, but this is the matter of our next investigations.

Acknowledgements. This work is the result of a cooperation supported by SIMILAR, European Network of Excellence (www.similar.cc). It has partially been supported and financed by France Telecom R&D.

References

1. Pavlovic, V., Sharma, R., Huang, T.S.: Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(7), 677–695 (1997)
2. Ong, S.C.W., Ranganath, S.: Automatic Sign Language Analysis: A Survey and the Future beyond Lexical Meaning. IEEE Transactions on Pattern Analysis and Machine Intelligence 27, 873–891 (2005)
3. Aran, O., Ari, I., Benoit, A., Campr, P., Carrillo, A.H., Fanard, F., Akarun, L., Caplier, A., Sankur, B.: SignTutor: An Interactive System for Sign Language Tutoring. IEEE Multimedia (accepted for publication, 2008)
4. Quost, B., Denoeux, T., Masson, M.-H.: Pairwise classifier combination using belief functions. Pattern Recognition Letters 28(5), 644–653 (2007)
5. Capelle, A.-S., Colot, O., Fernandez-Maloigne, C.: Evidential segmentation scheme of multi-echo MR images for the detection of brain tumors using neighborhood information. Information Fusion 5(3), 203–216 (2004)
6. Cornett, R.O.: Cued Speech, American Annals of the Deaf (1967)
7. Derpanis, K.G.: A review on vision-based hand gestures, internal report (2004)
8. Wu, Y., Huang, T.S.: Hand modeling, analysis, and recognition for vision based Human-Computer Interaction. IEEE Signal Processing Magazine 21, 51–60 (2001)

9. Burger, T., Aran, O., Caplier, A.: Modeling hesitation and conflict: A belief-based approach. In: Proc. ICMLA (2006)
10. Caplier, A., Bonnaud, L., Malassiotis, S., Strintzis, M.: Comparison of 2D and 3D analysis for automated Cued Speech gesture recognition. In: Proc. SPECOM, St Petersburg, Russia (2004)
11. Zhang, D., Lu, G.: Evaluation of MPEG-7 shape descriptors against other shape descriptors. Multimedia Systems 9(1) (2003)
12. Norkin, C.C., Levangie, P.K.: Joint structure and function, 2nd edn. F.A. Davis, Philadelphia (1992)
13. Shafer, G.: A Mathematical Theory of Evidence. Princeton University Press, Princeton (1976)
14. Smets, P., Kennes, R.: The transferable belief model. Artificial Intelligence 66(2), 191–234 (1994)
15. Boser, B., Guyon, I., Vapnik, V.: A training algorithm for optimal margin classifiers. In: Proc. Fifth Annual Workshop on Computational Learning Theory (1995)
16. Cortes, C., Vapnik, V.: Support-vector network. Machine Learning 20, 273–297 (1995)
17. Hsu, C.-W., Lin, C.-J.: A comparison of methods for multi-class support vector machines. IEEE Transactions on Neural Networks 13, 415–425 (2002)
18. Denoeux, T.: Modeling vague beliefs using fuzzy-valued belief structures. Fuzzy Sets and Systems 116(2), 167–199 (2000)
19. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines. Software (2001), `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
20. Aran, O., Burger, T., Caplier, A., Akarun, L.: A Belief-Based Sequential Fusion Approach for Fusing Manual and Non-Manual Signs. Pattern Recognition (accepted for publication, 2008)
21. Sudano, J.J.: Inverse pignistic probability transforms. In: Proc. Information Fusion (2002)
22. Cobb, B., Shenoy, P.: On the plausibility transformation method for translating belief function models to probability models. Int. J. Approx. Reasoning 41(3), 314–330 (2006)

Motion Feature Combination for Human Action Recognition in Video

Hongying Meng[1], Nick Pears[2], and Chris Bailey[2]

[1] Department of Computing and Informatics
University of Lincoln, Brayford Pool, Lincoln, LN6 7TS, U.K.
hmeng@lincoln.ac.uk
[2] Department of Computer Science, University of York, York, YO10 5DD, U.K.

Abstract. We study the human action recognition problem based on motion features directly extracted from video. In order to implement a fast human action recognition system, we select simple features that can be obtained from non-intensive computation. We propose to use the motion history image (MHI) as our fundamental representation of the motion. This is then further processed to give a histogram of the MHI and the Haar wavelet transform of the MHI. The combination of these two features is computed cheaply and has a lower dimension than the original MHI. The combined feature vector is tested in a Support Vector Machine (SVM) based human action recognition system and a significant performance improvement has been achieved. The system is efficient to be used in real-time human action classification systems.

Keywords: Event recognition, Human action recognition, Video analysis, Support Vector Machine.

1 Introduction

Event detection in video is becoming an increasingly important computer vision application, particularly in the context of activity classification [1]. Event recognition is an important goal for building intelligent systems which can react to what is going on in a scene. Event recognition is also a fundamental building block for interactive systems which can respond to gestural commands, instruct and correct a user learning athletics, gymnastics or dance movements, or interact with live actors in an augmented dance or theatre performance.

Recognizing actions of human actors from digital video is a challenging topic in computer vision with many fundamental applications in video surveillance, video indexing and social sciences. Feature extraction is the basis to perform many different tasks with video such as video object detection, object tracking and object classification.

Model based method are extremely challenging as there is large degree of variability in human behaviour. The highly articulated nature of the body leads to high dimensional models and the problem is further complicated by the non-rigid behaviour of clothing. Computationally intensive methods are needed for nonlinear modeling and optimisation. Recent research into anthropology has revealed that body dynamics are far more

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 151–163, 2008.

complicated than was earlier thought, affected by age, ethnicity, gender and many other circumstances [2].

Appearance-based models are based on the extraction of a 2D shape model directly from the images, to be classified (or matched) against a trained one. Motion-based models do not rely on static models of the person, but on human motion characteristics. Motion feature extraction and selection are two of the key components in these kinds of human action recognition systems.

In this paper, we study the human action classification problem based on motion features directly extracted from video. In order to implement fast human action recognition, we select simple features that can be obtained from non-intensive computation. In particular, we use the Motion History Image (MHI) [3] as our fundamental feature. We propose novel extraction methods to extract both spatial and temporal information from these initial MHI representations and we combine them as a new feature vector that has a lower dimension and provides better motion action information than the raw MHI information. This feature vector was used in a Support Vector Machine (SVM) based human action recognition system.

The rest of this paper is organised as follows: In section 2, we will give an introduction to some related work. In section 3, we give a brief overview of our system. In section 4, the detailed techniques of this system are explained including motion features, feature extraction methods and SVM classifier. In section 5, some experimental results are presented and compared. In section 6, the same combination idea has been tested on other features and significant improvement is also achieved. Finally, we give the conclusions.

2 Previous Work

Aggarwal and Cai [1] present an excellent overview of human motion analysis. Of the appearance based methods, template matching has increasingly gained attention. Bobick and Davis [3] use Motion Energy Images (MEI) and Motion History Images (MHI) to recognize many types of aerobics exercises. While their method is efficient, their work assumes that the actor is well segmented from the background and centred in the image.

Schuldt [4] proposed a method for recognizing complex motion patterns based on local space-time features in video and demonstrated such features can give good classification performance. They construct video representations in terms of local space-time features and integrate such representations with SVM classification schemes for recognition.

Ke [5] studies the use of volumetric features as an alternative to the local descriptor approaches for event detection in video sequences. They generalize the notion of 2D box features to 3D spatio-temporal volumetric features. They construct a real-time event detector for each action of interest by learning a cascade of filters based on volumetric features that efficiently scans video sequences in space and time. This event detector recognizes actions that are traditionally problematic for interest point methods such as smooth motions where insufficient space-time interest points are available. Their experiments demonstrate that the technique accurately detects actions on real-world sequences and is robust to changes in viewpoint, scale and action speed.

Weinland [6] introduces Motion History Volumes (MHV) as a free-viewpoint representation for human actions in the case of multiple calibrated, and background-subtracted, video cameras.

We note that the feature vector in these two methods is very expensive to construct and the learning process is difficult, because it needs a big data set for training.

Wong and Cipolla [7] proposed a new method to recognise primitive movements based on the Motion Gradient Orientation (MGO) image directly from image sequences. This process extracts the descriptive motion feature without depending on any tracking algorithms. By using a sparse Bayesian classifier, they obtained good classification results for human gesture recognition.

Ogata [8] proposed another efficient technique for human motion recognition based on motion history images and an eigenspace technique. In the proposed technique, they use Modified Motion History Images (MMHI) feature images and the eigenspace technique to realize high-speed recognition. The experiment results showed satisfactory performance of the technique. However, the eigenspace still needs to be constructed and sometimes this is difficult.

Recently, Dalal [9] proposed a Histogram of Oriented Gradient (HOG) appearance descriptors for image sequences and developed a detector for standing and moving people in video. In this work, several different motion coding schemes were tested and it was shown empirically that orientated histograms of differential optical flow give the best overall performance.

Oikonomopoulos [10] introduced a sparse representation of image sequences as a collection of spatiotemporal events that are localized at points that are salient both in space and time for human actions recognition.

These two methods need to detect salient points in the frames and then make suitable features for classification. This implies significant computational cost for detecting these points.

Meng [11] proposed a fast system for human action recognition which was based on very simple features. They chose MHI, MMHI, MGO and a linear classifier SVM for fast classification. Experimental results showed that this system could achieve good performance in human action recognition. Further, they [12] proposed to combine two kinds of motion features MHI and MMHI together and achieved better performance in human action recognition based on a linear SVM_2K classifier [13] [14]. However, both these systems could only work well in specific real-time applications with limited action classes because the overall performance on real-world challenging database were still not good enough.

3 Overall Architecture

We propose a novel architecture for fast human action recognition. In this architecture, a linear SVM was chosen and MHI provided our fundamental features. In contrast with the system in [11], we propose novel extraction methods to extract both spatial and temporal information from these initial MHI features and combine them in an efficient way as a new feature vector that has lower dimension and provides better motion action information than the raw MHI feature vector.

There are two reasons for choosing a linear SVM as the classifier in the system. Firstly SVM is a classifier that has achieved very good performance in lots of real-world classification problems. Secondly, SVM can deal with very high dimensional feature vectors, which means that there is plenty of freedom to choose the feature vectors. Finally the classifier is able to operate very quickly during the recognition process.

The overall architecture of the human action system is shown in figure 1. There are two parts in this system: a learning part and a classification part.

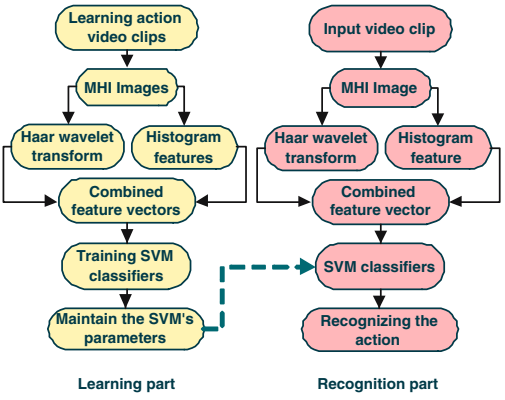


Fig. 1. SVM based human action recognition system. In the learning part, the combined feature vector of Haar wavelet transform and histogram of MHI were used for training a SVM classifier, and the obtained parameters were used in the recognition part.

The MHI feature vectors are obtained directly from human action video clips. The 2-D Haar wavelet transform was employed to extract spatial information within the MHI, while temporal information was extracted by computing the histogram of the MHI. Then these two feature vectors were combined to produce a lower dimensional and discriminative feature vector. Finally, the linear SVM was used for the classification process.

The learning part is processed using video data collected off-line. After that, the obtained parameters for the classifier can be used in a small, embedded computing device such as a field-programmable gate array (FPGA) or digital signal processor (DSP) based system, which can be embedded in the application and give real-time performance.

It should be mentioned here that, both 2-D Haar wavelet transform and histogram of the MHI are achieved with very low computational cost. We only keep the low-frequency part of the Haar wavelet transform. So the total dimension of the combined feature vector is lower than that of the original MHI feature.

4 Detail of the Method

In this section, we will give the detailed information of the key techniques used in our human action recognition system.

4.1 Motion Features

The recording of human actions usually needs large amounts of digital storage space and it is time consuming to browse the whole video to find the required information. It is also difficult to deal with this huge data in detection and recognition. Therefore, several motion features have been proposed to compact the whole motion sequence into one image to represent the motion. The most popular of these are the MHI, MMHI and MGO. These three motion features have the same size as the frame of the video, but they maintain the motion information within them. In [11], it has been found that MHI achieved best performance in classification tests across six categories of action sequence.

A motion history image (MHI) is a kind of temporal template. It is the weighted sum of past successive images and the weights decay as time lapses. Therefore, an MHI image contains past raw images within itself, where most recent image is brighter than past ones.

Normally, an MHI $H_\tau(u, v, k)$ at time k and location (u, v) is defined by the following equation 1:

$$H_\tau(u,v,k) = \begin{cases} \tau & if\ D(u,v,k) = 1 \\ max\{0, H_\tau(u,v,k) - 1\}, & otherwise \end{cases} \tag{1}$$

where $D(u, v, k)$ is a binary image obtained from subtraction of frames, and τ is the maximum duration a motion is stored. In general, τ is chosen as constant 255 where MHI can be easily represented as a grayscale image. An MHI pixel can have a range of values, whereas the Motion Energy Image (MEI) is its binary version. This can easily be computed by thresholding $H_\tau > 0$.

4.2 Histogram of MHI

The histogram of the MHI has bins which record the frequency at which each value (gray-level) occurs in the MHI, excluding the zero value, which does not contain any motion information of the action. Thus, typically we will have bins between 1 and 255 populated by one or more groupings, where each grouping of bins represents a motion trajectory. Clearly the most recent motion is at the right of the histogram, with the earliest motions recorded in the MHI being more toward the left of the histogram. The spread of each grouping in the histogram indicates the speed of the motion, with narrow groupings indicating fast motions and wide groupings indicating slow motions.

4.3 Haar Wavelet Transform

The Haar wavelet transform decomposes a signal into a time-frequency field based on the Haar wavelet function basis. For discrete digital signals, the discrete wavelet transform can be implemented efficiently by Mallat's fast algorithm [15]. The Mallat algorithm is in fact a classical scheme known in the signal processing community as a two-channel subband coder (see page 1 Wavelets and Filter Banks, by Strang and Nguyen [16]).

The Mallat algorithm is used for both wavelet decomposition and reconstruction. The algorithm has a pyramidal structure with the underlying operations being convolution and decimation. For a discrete signal $s = (s_0, s_1, \cdots, s_{N-1})(N = 2^L, L \in Z^+)$. For convenience, denote it as $c_{m,0} = s_m, m = 0, 1, \cdots, N-1$. Then Haar wavelet transform can be implemented by the following iteration:
For $l = 1, 2, \cdots, L$ and $m = 0, 1, \cdots, N/(l+1)$

$$\begin{cases} c_{m,l} = \sum_{k=0}^{1} h_k c_{k+2m,l-1} \\ d_{m,l} = \sum_{k=0}^{1} g_k c_{k+2m,l-1} \end{cases} \tag{2}$$

where the h_0, h_1 is low pass filter and g_0, g_1 is high pass filter:

$$h_0 = h_1 = \sqrt{2},$$
$$g_0 = \sqrt{2}, g_1 = -\sqrt{2}$$

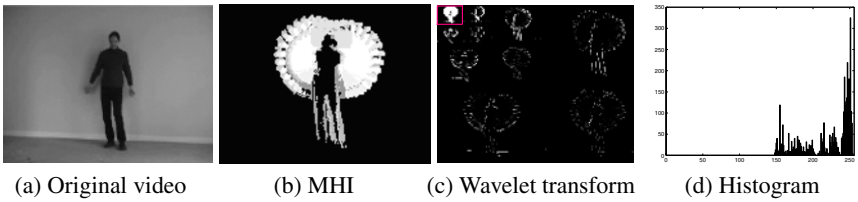
They are orthogonal:

$$g_k = (-1)^k h_{1-k} \tag{3}$$

and the obtained $\{c_{.,L}, d_{.,L}, d_{.,L-1}, \cdots, d_{.,1}\}$ is the discrete Haar wavelet transform of the signal.

An image is a 2-D signal and this 2-D space can be regarded as a separable space, which means that the wavelet transform on an image can be implemented using a 1-D wavelet transform. On the same level, it can be implemented on all the rows and then on all the columns.

In this paper, we only keep the low-frequency part of the Haar wavelet transform of the image. This part can represent the spatial information of the MHI very well in a lower dimension. The high-frequency information is more useful for representing edges, which is not really important in our system, and it is more susceptible to noise. Actually, this part can be implemented very quickly based on some specific algorithms.



(a) Original video (b) MHI (c) Wavelet transform (d) Histogram

Fig. 2. Motion feature of the action. (a) Original video (b)MHI (c) Haar wavelet transform of MHI (d) Histogram of MHI.

Figure 2 shows an example of a handwaving action. (a) is the original video clip, (b) is the MHI, (c) is the Haar wavelet transform of MHI where the red square is low frequency part and (d) is Histogram of the MHI.

4.4 Combining Features

The two feature vectors *histogram of MHI* and *Haar wavelet transform of MHI* are combined in the simplest way. The combined feature vector is built by concatenating these two feature vectors into a higher dimensional vector. In this way, the temporal and spatial information of the MHI are integrated into one feature vector while the dimension of the combined feature vector has lower dimension in comparison with MHI itself.

4.5 Support Vector Machine

SVM is a state-of-the-art classification technique with large application in a range of fields including text classification, face recognition and genomic classification, where patterns can be described by a finite set of characteristic features. We use the SVM for the classification component of our system. This is due to SVM being a classifier that has excellent performance on many real-world classification problems. Using arbitrary positive definite kernels provides a possibility to extend the SVM capability to handle high dimensional feature spaces.

Originally, the SVM is a binary classifier in a higher dimensional space where a maximal separating hyperplane is constructed. Two parallel hyperplanes are constructed on each side of the hyperplane that separates the data. The separating hyperplane is the hyperplane that maximizes the distance between the two parallel hyperplanes. If we have a training dataset $\left\{\mathbf{x}_i | \mathbf{x}_i \in R^d\right\}$, and its binary labels are denoted as $\{y_i | y_i = \pm 1\}$, the norm-2 soft-margin SVM can be represented as a constrained optimization problem

$$\min_{w,b,\xi} \frac{1}{2}||\mathbf{w}||^2 + C\sum_i \xi_i \tag{4}$$

s.t.

$$\begin{gathered}\langle \mathbf{x}_i, \mathbf{w}\rangle + b \geq 1 - \xi_i,\ y_i = 1,\\ \langle \mathbf{x}_i, \mathbf{w}\rangle + b \leq -1 + \xi_i,\ y_i = -1,\\ \xi_i \geq 0,\end{gathered}$$

where C is a penalty parameter and ξ_i are slack variables. The vector $\mathbf{w} \in R^d$ points perpendicular to the separating hyperplane. Adding the offset parameter b allows us to increase the margin. It can be converted by applying Lagrange multipliers into its Wolfe dual problem and can be solved by quadratic programming methods.

The primal optimum solution for weight vector $\mathbf{w}$ can be represented as

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i. \tag{5}$$

where $0 \leq \alpha_i \leq C$. Obviously, $\mathbf{w}$ can be expressed as a linear combination of the support vectors for which $\alpha_i > 0$. For a testing feature vector $\mathbf{x}$, the decision function η and its estimated label h are:

$$h(\mathbf{x}) = sign(\eta(\mathbf{x})) = sign(\langle \mathbf{w}, \mathbf{x}\rangle + b). \tag{6}$$

The original optimal hyperplane algorithm was a linear classifier. However, many researchers have created non-linear classifiers by applying a kernel trick [17] and thus the SVM can be generalized to the case where the decision function is a non-linear function of the data.

Multiclass SVMs are usually implemented by combining several two-class SVMs. In each binary SVM, only one class is labelled as "1" and the others labelled as "-1". The one-versus-all method uses a winner-takes-all strategy.

If there are M classes, then the SVM method will construct M binary classifiers by learning. During the testing process, each classifier will get a confidence coefficient $\{\eta_j(\mathbf{x}) | j = 1, 2, \cdots, M\}$ and the class k with the maximum confidence coefficient will be assigned to this sample $\mathbf{x}$.

$$h(\mathbf{x}) = k, \qquad if \;\; \eta_k(\mathbf{x}) = max_{j=1}^{M}(\eta_j(\mathbf{x})). \tag{7}$$

Our human action recognition problem here is a multi-class classification case. If, for example, we have six classes, then six SVM classifiers are trained based on motion features such as the MHI obtained from human action video clips in a training dataset. For each SVM training, one class is labeled as "1" and the rest classes are labeled as "-1". After the training, each SVM classifier is represented by two parameters $\mathbf{w}$ and b. These parameters will be stored in the internal memory of the FPGA. In the recognition process, one inner product between obtained MHI and $\mathbf{w}$ will be calculated and added to b for each SVM classifier. Then the final predicted label for the action video will go to the class with the maximum one in the computed six values.

5 Experimental Results

5.1 Dataset

For the evaluation, we use a challenging human action recognition database, recorded by Christian Schuldt [4]. It contains six types of human actions (walking, jogging, running, boxing, hand waving and hand clapping) performed several times by 25 subjects in four different scenarios: outdoors (s1), outdoors with scale variation (s2), outdoors with different clothes (s3) and indoors (s4).

This database contains 2391 sequences. All sequences were taken over homogeneous backgrounds with a static camera with 25Hz frame rate. The sequences were down-sampled to the spatial resolution of 160×120 pixels. For all the action sequences, the length of the sequences are vary and the average is four seconds (about 100 frames). To the best of our knowledge, this is the largest video database with sequences of human actions taken over different scenarios. All sequences were divided with respect to the subjects into a training set (8 persons), a validation set (8 persons) and a test set (9 persons). In our experiment, the classifiers were trained on the training set while classification results were obtained on the test set.

Figure 3 showed six types of human actions in the database: walking, jogging, running, boxing, handclapping and handwaving. Row (a) are the original videos, (b) and (c) are the associated MHI and Histogram of MHI features.

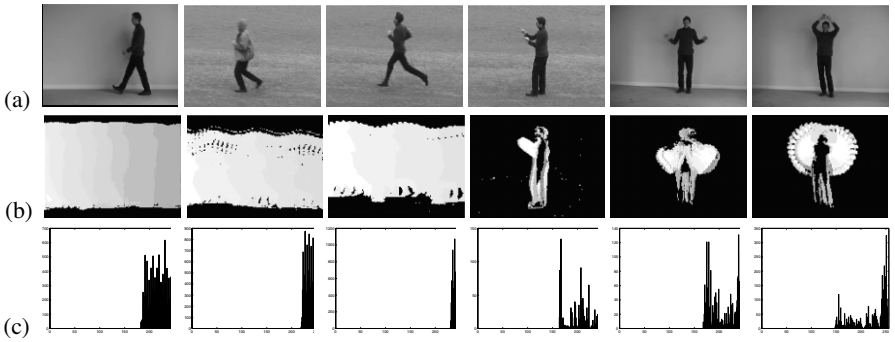


Fig. 3. Six types of human actions in the database: walking, jogging, running, boxing, hand-clapping and handwaving. Row (a) are the original videos, (b) and (c) are associate MHI and Histogram of MHI features.

5.2 Experimental Setup

Our experiments were carried out on all four different scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. In the same manner as paper [5], each sequence is treated individually during the training and classification process. In all the following experiments, the parameters were chosen to be the same. The threshold in differential frame computing was chosen as 25 and τ was chosen as constant 255 for MHI construction.

A MHI was calculated from each action sequence with about 100 frames. The size of each MHI is $160 \times 120 = 19200$, which is same width as that of the frames in the videos. The values of MHI are in the interval of $[0, 255]$. Then each MHI was decomposed using a 2-D Haar wavelet transform to $L = 3$ levels. Thus the size of the low frequency part of the Haar wavelet transform of MHI is $20 \times 15 = 300$. Since the length of the histogram of MHI is 255, the length of combined feature vector is 555.

In our system, each SVM was trained based on features obtained from human action video clips in a training dataset. These video clips have their own labels such as "walking," "running" and so on. In classification, we actually get a six-class classification problem. The SVM training can be implemented using programs freely available on the web, such as SVM^{light} [18]. Finally, we obtained several SVM classifiers with associated parameters.

In the recognition process, feature vectors will be extracted from the input human action video sample. Then all the SVM classifiers obtained from the training process will classify the extracted feature vector. Finally, the class with maximum confidence coefficient within these SVM classifiers will be assigned to this sample.

5.3 Experiment Results

Tables 1 show the classification confusion matrix based on the method proposed in paper [5].The confusion matrices show the motion label (vertical) versus the classification results (horizontal). Each cell (i, j) in the table shows the percentage of class

Table 1. Ke's confusion matrix, trace=377.8

	Walk	Jog	Run	Box	Clap	Wave
Walk	**80.6**	11.1	8.3	0.0	0.0	0.0
Jog	30.6	**36.2**	33.3	0.0	0.0	0.0
Run	2.8	25.0	**44.4**	0.0	27.8	0.0
Box	0.0	2.8	11.1	**69.4**	11.1	5.6
Clap	0.0	0.0	5.6	36.1	**55.6**	2.8
Wave	0.0	5.6	0.0	2.8	0.0	**91.7**

Table 2. MHI_S's confusion matrix, trace=377.7

	Walk	Jog	Run	Box	Clap	Wave
Walk	**56.9**	18.1	22.2	0.0	0.0	2.8
Jog	45.1	**29.9**	22.9	1.4	0.0	0.7
Run	34.7	27.8	**36.1**	0.0	0.0	1.4
Box	0.0	0.0	0.0	**89.5**	2.1	8.4
Clap	0.0	0.0	0.0	5.6	**88.9**	5.6
Wave	0.0	0.0	0.0	12.5	11.1	**76.4**

Table 3. MHI_hist's confusion matrix, trace=328.6

	Walk	Jog	Run	Box	Clap	Wave
Walk	**62.5**	32.6	0.0	1.4	1.4	2.1
Jog	12.5	**58.3**	25.0	0.0	0.0	4.2
Run	0.7	18.8	**77.1**	0.0	0.0	3.5
Box	4.9	2.8	0.7	**17.5**	61.5	12.6
Clap	4.9	2.1	0.7	11.1	**75.0**	6.3
Wave	5.6	3.5	6.9	20.1	25.7	**38.2**

i action being recognized as class j. Then trace of the matrices show the percentage of the correctly recognized action, while the remaining cells show the percentage of misclassification.

In order to study the performance of the Haar wavelet transform of MHI and histogram of MHI, we used linear SVM classifier on them separately and compared their performance. Table 2 and table 3 shows the confusion matrix obtained for Haar wavelet transform and histogram of MHI separately. From these two tables, it can be seen that Haar wavelet transform of MHI obtains a similar performance to Ke's method. This feature did very well in distinguishing the last three groups. On the other hand, histogram of MHI did not do well on overall performance. But it has the power to distinguish the first three groups. That demonstrates that they keep different information from MHI.

Table 4. MHI_S&MHI_hist's confusion matrix, trace=425.6

	Walk	Jog	Run	Box	Clap	Wave
Walk	**68.8**	11.1	17.4	0.0	0.0	2.8
Jog	36.8	**36.1**	25.0	1.4	0.0	0.7
Run	14.6	20.1	**63.9**	0.0	0.0	1.4
Box	0.0	0.0	0.0	**89.5**	2.1	8.4
Clap	0.0	0.0	0.0	4.9	**89.6**	5.6
Wave	0.0	0.0	0.0	11.1	11.1	**77.8**

Table 4 show the confusion matrix obtained from our system in which combined feature were used. From this table, we can see that the overall performance has got a significant improvement on Ke's method based on volumetric features. Good performance is achieved in distinguishing all of the six actions in the dataset.

It should be mentioned here that in paper [4], the performance is slightly better where trace=430.3. But our system was trained in the same way as [5] to detect a single instance of each action within arbitrary sequences while Schuldt's system has the easier task of classifying each complete sequence (containing several repetitions of same action) into one of six classes.

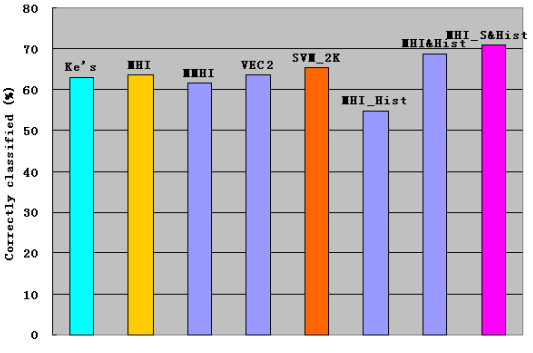


Fig. 4. Comparison results on the correctly classified rate based on different methods: Ke's method; SVM on MHI; SVM on MMHI; SVM on the concatenated feature (VEC2) of MHI and MMHI and SVM_2K on MHI and MMHI;SVM on histogram of MHI; SVM on the combined feature of MHI and histogram of MHI; SVM on combined feature of Haar wavelet transform of MHI and histogram of MHI.

We also compared the correctly classified rate based on our system with other previous results in the figure 4. The first one is the Ke's method, the second, third and sixth are SVM based on individual features MHI, MMHI and Histogram of MHI respectively. The fourth one is SVM based on combined feature from MHI and MMHI. The fifth is using SVM_2K classifier on both MHI and MMHI. The seventh is SVM on combined feature from MHI and its histogram. The last one is the results SVM based on Haar wavelet transform of MHI and histogram of MHI. This last result achieves the best overall performance of approximately 71% correct classification.

6 Extension of the Idea

In the previous sections, we combined two different types features extracted from same MHI feature and achieved significant improvement on the performance. The reason is that these two features extract different characteristics of the motion feature. In fact, this idea can be further extended to combine different types of features extracted from different motion features. In [19], we combined the histogram of MHI with Motion Geometric Distribution (MGD) feature vector extracted from the Motion History Histogram. The main common point between MGD feature and Haar wavelet transform feature is that both of them represented spatial information of the motion features. Table 5 showed the experiment results on the same dataset. It achieves the best overall performance of above 80% correct classification.

Table 5. MGD & Hist. of MHI's confusion matrix, trace=481.9

	Walk	Jog	Run	Box	Clap	Wave
Walk	**66.0**	31.3	0.0	0.0	2.1	0.7
Jog	13.9	**62.5**	21.5	1.4	0.0	0.7
Run	2.1	16.7	**79.9**	0.0	0.0	1.4
Box	0.0	0.0	0.0	**88.8**	2.8	8.4
Clap	0.0	0.0	0.0	3.5	**93.1**	3.5
Wave	0.0	0.0	0.0	1.4	6.9	**91.7**

7 Conclusions

In this paper, we proposed a system for fast human action recognition. Potential applications include security systems, man-machine communication, and ubiquitous vision systems. The proposed method does not rely on accurate tracking as many other works do, since many tracking algorithms incur a prohibitive computational cost for the system. Our system is based on simple features in order to achieve high-speed recognition, particularly in real-time embedded vision applications.

In comparison with local SVM methods by Schuldt [4], our feature vector is much easier to obtain because we don't need to find interest points in each frame. We also don't need a validation dataset for parameter tuning.

In comparison with Meng's [11] [12] methods, we use a Haar wavelet transform and histogram methods to build a new feature vector from the MHI representation. This new feature vector contains the important information of the MHI and also has a lower dimension. Experimental results demonstrate that these techniques made a significant improvement on the human action recognition performance compared to other methods.

If the learning part of the system is conducted off-line, this system has great potential for implementation in small, embedded computing devices, typically FPGA or DSP based systems, which can be embedded in the application and give real-time performance.

Acknowledgements. This work is supported by the DTI (UK) and Broadcom Ltd.

References

1. Aggarwal, J.K., Cai, Q.: Human motion analysis: a review. Comput. Vis. Image Underst. 73, 428–440 (1999)
2. Farnell, B.: Moving bodies, acting selves. Annual Review of Anthropology 28, 341–373 (1999)
3. Bobick, A.F., Davis, J.W.: The recognition of human movement using temporal templates. IEEE Trans. Pattern Anal. Mach. Intell. 23, 257–267 (2001)
4. Schuldt, C., Laptev, I., Caputo, B.: Recognizing human actions: a local SVM approach. In: Proc. Int. Conf. Pattern Recognition (ICPR 2004), Cambridge, U.K (2004)
5. Ke, Y., Sukthankar, R., Hebert, M.: Efficient visual event detection using volumetric features. In: Proceedings of International Conference on Computer Vision, Beijing, China, October 15-21, pp. 166–173 (2005)
6. Weinland, D., Ronfard, R., Boyer, E.: Motion history volumes for free viewpoint action recognition. In: IEEE International Workshop on modeling People and Human Interaction (PHI 2005) (2005)
7. Wong, S.F., Cipolla, R.: Real-time adaptive hand motion recognition using a sparse bayesian classifier. In: ICCV-HCI, pp. 170–179 (2005)
8. Ogata, T., Tan, J.K., Ishikawa, S.: High-speed human motion recognition based on a motion history image and an eigenspace. IEICE Transactions on Information and Systems E89, 281–289 (2006)
9. Dalal, N., Triggs, B., Schmid, C.: Human detection using oriented histograms of flow and appearance. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 428–441. Springer, Heidelberg (2006)
10. Oikonomopoulos, A., Patras, I., Pantic, M.: Kernel-based recognition of human actions using spatiotemporal salient points. In: Proceedings of IEEE Int'l Conf. on Computer Vision and Pattern Recognition 2006, vol. 3 (2006)
11. Meng, H., Pears, N., Bailey, C.: Recognizing human actions based on motion information and svm. In: 2nd IET International Conference on Intelligent Environments, Athens, Greece, IET, pp. 239–245 (2006)
12. Meng, H., Pears, N., Bailey, C.: Human action classification using svm_2k classifier on motion features. In: Gunsel, B., Jain, A.K., Tekalp, A.M., Sankur, B. (eds.) MRCS 2006. LNCS, vol. 4105, pp. 458–465. Springer, Heidelberg (2006)
13. Meng, H., Shawe-Taylor, J., Szedmak, S., Farquhar, J.D.R.: Support vector machine to synthesise kernels. In: Deterministic and Statistical Methods in Machine Learning, 242–255 (2004)
14. Farquhar, J.D.R., Hardoon, D.R., Meng, H., Shawe-Taylor, J., Szedmak, S.: Two view learning: Svm-2k, theory and practice. In: NIPS (2005)
15. Mallat, S.: A theory for multiresolution signal decomposition: the wavelet representation. IEEE Transactions on Pattern Analysis and Machine Intelligence 11, 674–693 (1989)
16. Strang, G., Nguyen, T.: Wavelets and Filter Banks. Wellesley Cambridge Press (1996)
17. Aizerman, A., Braverman, E.M., Rozoner, L.I.: Theoretical foundations of the potential function method in pattern recognition learning. Automation and Remote Control 25, 821–837 (1964)
18. Joachims, T.: Making large-scale svm learning practical. In: Oikonomopoulos, A., Patras, I., Pantic, M. (eds.) Advances in Kernel Methods - Support Vector Learning, USA. MIT-Press, Cambridge (1999)
19. Meng, H., Pears, N., Bailey, C.: A human action recognition system for embedded computer vision application. In: The 3rd IEEE workshop on Embeded Computer Vision, Minneapolis,USA (2007)

Optimal Factor Analysis and Applications to Content-Based Image Retrieval

Yuhua Zhu[1], Washington Mio[2], and Xiuwen Liu[1]

[1] Dept. of Computer Science, Florida State University, Tallahassee FL 32306, U.S.A.
[2] Dept. of Mathematics, Florida State University, Tallahassee FL 32306, U.S.A.

Abstract. We formulate and develop computational strategies for *Optimal Factor Analysis* (OFA), a linear dimension reduction technique designed to learn low-dimensional representations that optimize discrimination based on the nearest-neighbor classifier. The methods are applied to content-based image categorization and retrieval using a representation of images by histograms of their spectral components. Various experiments are carried out and the results are compared to those that have been previously reported for some other image retrieval systems.

Keywords: Linear dimension reduction, image classification, content-based image retrieval, optimal factor analysis.

1 Introduction

We develop *Optimal Factor Analysis* (OFA), a linear dimension-reduction technique that optimizes the discriminative ability of the nearest-neighbor classifier for a given data classification problem. We apply the technique to content-based categorization and retrieval of images using a representation based on the statistics of their spectral components. This investigation is motivated by the need to develop intelligent and scalable systems capable of indexing and retrieving images from large and complex image libraries in an automated manner. Classical approaches based on "expert" annotations are not viable for large data sets.

For the image categorization problem, we shall assume that a training database of labeled images representing various different classes of objects is available and the goal is to learn optimal low-dimensional features or "signatures" to assign a query image to the correct class. In content-based image retrieval, one of the objectives is to find the top ℓ matches in a database to a query image, where the number ℓ is prescribed by the user. In the proposed approach, categorization and retrieval are closed related. We use a categorization algorithm to organize an entire database according to features learned from a training set. Given a query image I, we first rank the classes using the nearest neighbor classifier applied to the learned low-dimensional features and then retrieve images sequentially starting from the top ranked class.

The problem of classifying images in a database into semantic categories arises in many different levels of generality. For example, the problem can be as broad as separating images that depict an indoor or outdoor scene, or it may involve much more specific categorization into classes such as cars, people, and flowers. As the breadth of the semantic categories may vary considerably, the development of general strategies poses

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 164–176, 2008.

significant challenges. This motivated us to approach the problem in two stages. First, we extract "stable" features that are able to capture information about the structure and semantic content of an image. Subsequently, we employ learning techniques to identify the factors that have the highest discriminating power for a particular classification problem.

The histogram of an image is useful, however, it tends to have limited discriminating power because it contains little information about the finer structure of an image. To remedy the situation, we use histograms of multiple spectral components, as they retain a significant amount of information about texture patterns and edges. The statistics of spectral components have been studied in the past primarily in the context of texture analysis and synthesis. In [11], it is demonstrated that marginal distributions of spectral components suffice to characterize homogeneous textures; other studies include [5] and [9]. To provide some preliminary evidence of the suitability of spectral histogram (SH) features, in Section 3, we report the results of a retrieval experiment on a database of 1,000 images representing 10 different semantic categories. The relevance of an image is determined by the nearest-neighbor criterion applied to a number of SH-features combined into a single feature vector. Even without a learning component, we already observe a performance comparable to those exhibited by some existing retrieval systems.

Optimal Factor Analysis will be employed with a twofold purpose: (a) to identify and split off the most discriminating factors of the SH-features; (b) to lower the dimension of the representation to reduce complexity and improve computational efficiency. A preliminary form of OFA was introduced in [3] as Splitting Factor Analysis. Given a (small) positive integer k, the goal of OFA is to find an "optimal" k-dimensional linear reduction of the original image features for a particular categorization or indexing problem. Image categorization and retrieval will be based on the nearest neighbor classifier applied to the reduced features, as explained in more detail below. We employ OFA in the context of SH-features, but it will be presented in a more general feature learning framework.

Image retrieval strategies employing a variety of methods have been investigated in [8], [1], [6], [7], [10], [2]. Further references can be found in these papers. Some of these proposals employ a relevance feedback mechanism in an attempt to progressively improve the quality of retrieval. Although not discussed in this paper, a feedback component can be incorporated to the proposed strategy by gradually adding to the training set images for which the quality of retrieval was low.

The paper is organized as follows. In Section 2, we describe the histogram features that will be used to characterize image content. Preliminary retrieval experiments using these features are described in Section 3. Section 4 contains a discussion of Optimal Factor Analysis, and Sections 6 and 7 are devoted to applications of the machine learning methodology to image categorization and retrieval. Section 8 closes the discussion with a summary and a few remarks on refinements of the proposed methods.

2 Spectral Histogram Features

Let I be a gray-scale image and F a convolution filter. The spectral component I_F of I associated with F is the image I_F obtained through the convolution of I and F, which

is given at pixel location p by

$$I_F(p) = F * I(p) = \sum_q F(q)\, I(p-q), \tag{1}$$

where the summation is taken over the pixels of F. For a color image, we apply the filter to its R,G,B channels. For a given set of bins, which will be assumed fixed throughout the paper, we let $h(I, F)$ denote the corresponding histogram of I_F. We refer to $h(I, F)$ as the spectral histogram (SH) feature of the image I associated with the filter F. If the number of bins is b, the SH-feature $h(I, F)$ can be viewed as a vector in $\mathbb{R}^b$. Figure 1 illustrates the process of obtaining SH-features. Frames (a) and (b) show a color image and its red channel response to a Laplacian filter, respectively. The last panel shows the 11-bin histogram of the filtered image.

(a) (b) (c)

Fig. 1. (a) An image; (b) the red-channel response to a Laplacian filter; (c) the associated 11-bin histogram

If $\mathcal{F} = \{F_1, \ldots, F_r\}$ is a bank of filters, the SH-features associated with the family $\mathcal{F}$ is the collection $h(I, F_i)$, $1 \leqslant i \leqslant r$, combined into the single m-dimensional vector

$$h(I, \mathcal{F}) = (h(I, F_1), \ldots, h(I, F_r)), \tag{2}$$

where $m = rb$. For a color image, $m = 3rb$. Banks of filters used in this paper consist of Gabor filters of different widths and orientations, gradient filters, and Laplacian of Gaussians.

3 SH-Features for Image Retrieval

To offer some preliminary evidence that image representation by SH-features may be attractive for retrieval, we perform a simple retrieval experiment using the Euclidean distance between histograms. Even without a learning component, the results are already comparable to those obtained with some existing systems. To compare the results objectively with those reported in [8] for SIMPLIcity and color histograms, we use the same subset of the Corel data set consisting of 10 semantic categories, each with 100 images. We refer to this data set as Corel-1000. The categories are as follows: (1) African people and villages; (2) beach scenes; (3) buildings; (4) buses; (5) dinosaurs; (6) elephants; (7) flowers; (8) horses; (9) mountains and glaciers; (10) food. Three samples from each category are shown in Figure 2. The examples are emblematic of the large variations observed even within a semantic category.

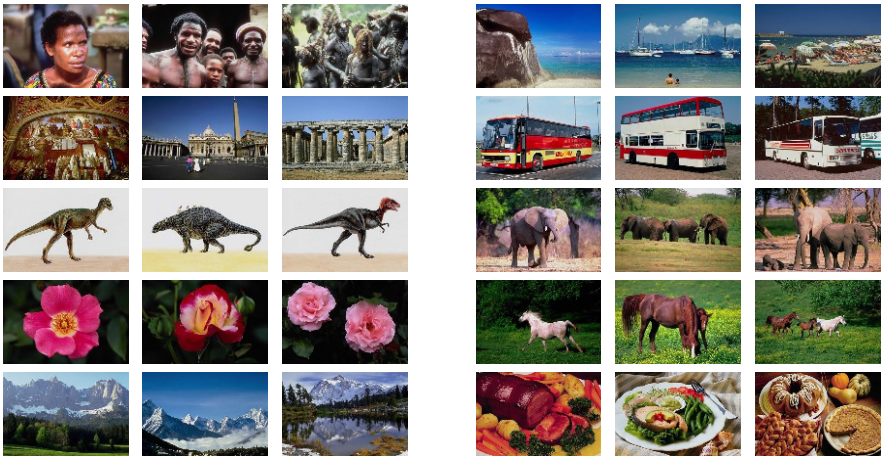


Fig. 2. Samples from Corel-1000: three images from 10 classes, each consisting of 100 images

We utilize a bank of 5 filters and apply each filter to the R, G, and B channels of the images to obtain a total of 15 histograms per image. Each histogram consists of 11 bins so that the SH-feature vector $h(I, \mathcal{F})$ has dimension 165. For a query image I from the database, we calculate the Euclidean distances between $h(I, \mathcal{F})$ and $h(J, \mathcal{F})$, for every J in the database, and rank the images according to increasing distances. For comparison purposes, as in [8], we calculate the weighted precision and the average rank, which are defined as follows. The retrieval precision for the top ℓ returns, is n_ℓ/ℓ, where n_ℓ is the number of correct matches. The weighted precision for a query image I is

$$p(I) = \frac{1}{100} \sum_{\ell=1}^{100} \frac{n_\ell}{\ell}. \tag{3}$$

For a query image I, rank order all 1,000 images in the database, as described above. The average rank $r(I)$ is the mean value of the ranks of all images that belong to the same class as I. Figures 3(a) and 3(b) show the mean values

$$\bar{p}_i = \frac{1}{100} \sum_{I \in C_i} p(I) \quad \text{and} \quad \bar{r}_i = \frac{1}{100} \sum_{I \in C_i} r(I), \tag{4}$$

of the weighted precision and average rank within each class C_i, $1 \leqslant i \leqslant 10$. High mean precision and low mean rank reflect high retrieval performance. The results obtained with SH-features are compared to those reported in [8] for SIMPLIcity and for color histograms with the earth mover's distance (EMD) investigated in [6]. In Figure 3, color histograms 1 and 2 refer to EMD applied to histograms with a different number of bins.

4 Optimal Factor Analysis

We develop Optimal Factor Analysis (OFA), a linear feature learning technique whose goal is to find a linear mapping that reduces the dimension of data representation while

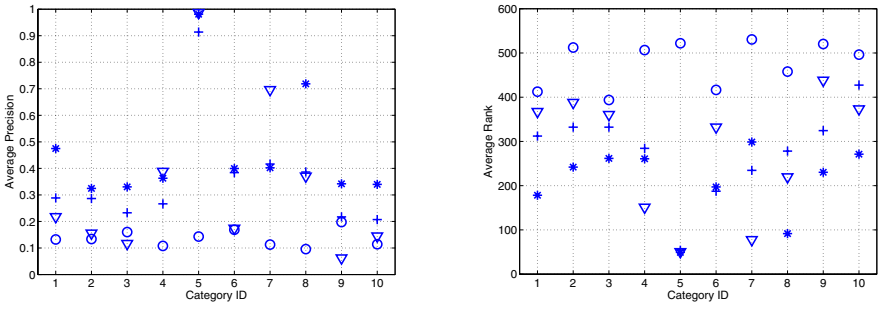


Fig. 3. (a) Plots of $\bar{p}_i$ and $\bar{r}_i$, $1 \leqslant i \leqslant 10$. The methods are labeled as: ($\triangledown$) spectral histogram; ($*$) SIMPLIcity; ($\circ$) color histogram 1; (+) color histogram 2.

optimizing the discriminative ability of the nearest neighbor classifier, as measured by its performance on training data. We assume that the training set is formed by feature vectors in Euclidean space $\mathbb{R}^m$ and consists of labeled representatives from P different classes of objects. For each integer c, $1 \leq c \leq P$, we denote the vectors in class c by $x_{c,1}, \ldots, x_{c,t_c}$.

If $A\colon \mathbb{R}^m \to \mathbb{R}^k$ is a linear transformation, the quantity

$$\rho(x_{c,i}; A) = \frac{\min_{c \neq b,j} \|Ax_{c,i} - Ax_{b,j}\|^p}{\min_{j \neq i} \|Ax_{c,i} - Ax_{c,j}\|^p + \epsilon} \tag{5}$$

provides a measurement of how well the nearest-neighbor classifier applied to the reduced data identifies the element represented by $x_{c,i}$ as belonging to class c. Here, $\epsilon > 0$ is a small number used to prevent vanishing denominators and $p > 0$ is an exponent that can be adjusted to regularize ρ in different ways. In this paper, we set $p = 2$. A large value of $\rho(x_{c,i}; A)$ indicates that, after the transformation A is applied, $x_{c,i}$ lies much closer to its own class than to other classes. A value $\rho(x_{c,i}; A) \approx 1$ indicates a transition between correct and incorrect decisions by the nearest neighbor classifier. The function ρ is similar to that used in the development of Optimal Component Analysis (OCA) [4]. Note that expression (5) can be easily modified to reflect the performance of the K-nearest neighbor classifier.

The idea is to choose a transformation A that maximizes the average value of $\rho(x_{c,i}; A)$ over the training set. To control bias with respect to particular classes, we scale $\rho(x_{c,i}; A)$ with a sigmoid of the form

$$\sigma(x) = \frac{1}{1 + e^{-\beta x}} \tag{6}$$

before taking the average. We identify linear maps $A\colon \mathbb{R}^m \to \mathbb{R}^k$ with $k \times m$ matrices, in the usual way, and define a performance function $F\colon \mathbb{R}^{k \times m} \to \mathbb{R}$ by

$$F(A) = \frac{1}{P} \sum_{c=1}^{P} \left(\frac{1}{t_c} \sum_{i=1}^{t_c} \sigma\left(\rho(x_{c,i}; A) - 1\right) \right). \tag{7}$$

Scaling an entire dataset does not change decisions based on the nearest-neighbor classifier. This is reflected in the fact that F is (nearly) scale invariant; that is, $F(A) \approx F(rA)$, for $r > 0$. Equality does not hold exactly because $\epsilon > 0$, but in practice, ϵ is negligible. Thus, we fix the scale and optimize F over matrices A of unit Frobenius norm. Let

$$\mathbb{S} = \left\{ A \in \mathbb{R}^{k \times m} : \ \|A\|^2 = \text{tr}\,(AA^T) = 1 \right\} \tag{8}$$

be the unit sphere in $\mathbb{R}^{k \times m}$. The goal of OFA is to maximize the performance function F over $\mathbb{S}$; that is, to find

$$\hat{A} = \underset{A \in \mathbb{S}}{\text{argmax}}\, F(A). \tag{9}$$

Due to the existence of multiple local maxima of F, the numerical estimation of $\hat{A}$ is carried out with a stochastic gradient search. We remark that this optimization problem is simpler than the corresponding problem for OCA because the OFA search is performed over a sphere instead of a Grassmann manifold. While OCA only considers dimension reduction via orthogonal projections to k-dimensional subspaces of $\mathbb{R}^m$, OFA allows more general linear mappings. Thus, OFA may produce k-dimensional features more effective for classification with significant computational gains.

5 Estimating $\hat{A}$

Our computational approach to the estimation of $\hat{A}$ is based on simulated annealing and is similar to the strategy adopted by Liu *et al.* for OCA [4]. We begin with the details of a deterministic gradient search for maxima of F over the unit sphere $\mathbb{S}$ and then outline the routine changes needed to carry out a stochastic search using simulated annealing with a Metropolis-Hastings acceptance-rejection criterion.

5.1 Deterministic Gradient

Given $A \in \mathbb{S}$, to estimate the gradient vector field $\nabla_{\mathbb{S}} F$ on $\mathbb{S}$ associated with the performance function F, we first calculate $\nabla F(A)$, the gradient of F viewed as a function on $\mathbb{R}^{k \times m}$. Since F is nearly scale invariant,

$$\nabla F(A) \approx \nabla_{\mathbb{S}} F(A), \tag{10}$$

as the component of $\nabla F(A)$ normal to the sphere is almost negligible. The numerical estimation of the left-hand side of (10) only involves standard procedures. For $1 \leqslant i \leqslant k$, $1 \leqslant j \leqslant m$, let E_{ij} be the $k \times m$ matrix whose (i, j) entry is 1 and all others vanish. The partial derivative of F in the direction E_{ij} is estimated as

$$\partial_{ij} F(A) \approx \frac{F(A + \delta E_{ij}) - F(A)}{\delta},$$

with $\delta > 0$ small. Then, $\nabla F(A)$ can be approximated by

$$\bar{\nabla} F(A) = \sum_{i,j} \partial_{ij} F(A) E_{ij}. \tag{11}$$

The vector $\bar{\nabla} F(A)$ is nearly tangential to $\mathbb{S}$ at A. We enforce full tangentiality and obtain a more accurate estimation of $\nabla_{\mathbb{S}} F(A)$ by subtracting the component normal to the sphere $\mathbb{S}$, as follows. For any $A \in \mathbb{S}$, the outer unit normal vector to $\mathbb{S}$ at A in $\mathbb{R}^{k \times m}$ is A itself. Thus, we adopt the estimate

$$\nabla_{\mathbb{S}} F(A) \approx \bar{\nabla} F(A) - \left\langle \bar{\nabla} F(A), A \right\rangle A. \tag{12}$$

A deterministic gradient search for (local) maxima of F on $\mathbb{S}$ can be carried out with the following algorithm.

Algorithm: Deterministic Gradient Search

1. Choose a threshold value $\epsilon > 0$ and a step size $\delta > 0$.
2. Initialize the search with some $A \in \mathbb{S}$.
3. Calculate $\nabla_{\mathbb{S}} F(A)$ using Eqns. 11 and 12.
4. If $\|\nabla_{\mathbb{S}} F(A)\| < \epsilon$, set $\hat{A} = A$ and stop. Else, update A according to

$$A = A \cos\left(\delta \|\nabla_{\mathbb{S}} F(A)\|\right) + \frac{\nabla_{\mathbb{S}} F(A)}{\|\nabla_{\mathbb{S}} F(A)\|} \sin\left(\delta \|\nabla_{\mathbb{S}} F(A)\|\right).$$

5. Go to Step 3.

Remarks:

(a) The update of A described in Step 4 of the algorithm has the effect of displacing A by $\delta \|\nabla_{\mathbb{S}} F(A)\|$ units of length along the great circle of $\mathbb{S}$ through A in the direction $\nabla_{\mathbb{S}} F(A)$.
(b) We often initialize the search with a linear mapping obtained from classical dimension reduction techniques such as principal component analysis or linear discriminant analysis.

5.2 Stochastic Search

Our next goal is to add a stochastic component to the deterministic gradient field $\nabla_{\mathbb{S}} F$ on $\mathbb{S}$. To simplify the calculation, instead of considering stochastic processes on the sphere, we first add a random component to $\nabla_{\mathbb{S}} F(A)$ as a vector in $\mathbb{R}^{k \times m}$ and then project it to the tangent space to $\mathbb{S}$ at A. We adopt the notation $\Pi_A \colon \mathbb{R}^{m \times k} \to T_A \mathbb{S}$ for the orthogonal projection of $\mathbb{R}^{m \times k}$ onto the tangent space of $\mathbb{S}$ at A, which is given by $\Pi_A(X) = X - \langle X, A \rangle A$.

Algorithm: Stochastic Gradient Search

1. Choose $A \in \mathbb{S}$, a cooling ratio $\gamma > 1$, an initial temperature $T_0 > 0$, a step size $\delta > 0$, and a positive integer N to control the number of iterations.
2. Set $t = 0$ and initialize the search with $A_t = A \in \mathbb{S}$.
3. Calculate $\nabla_{\mathbb{S}} F(A_t)$ using Eqns. 11 and 12.

4. Generate samples $w_{ij}(t) \in \mathbb{R}$, $1 \le i \le m$, $1 \le j \le k$, from the standard normal distribution and construct the tangent vector

$$f(t) = \delta \nabla_{\mathbb{S}} F(A_t) + \sqrt{2\delta T_t}\, \Pi_{A_t} \left(\sum_{i,j} w_{ij}(t) E_{ij} \right).$$

5. Moving along an arc of length $\|f(t)\|$ on the great circle through A_t in the direction of $f(t)$, define a candidate $B \in \mathbb{S}$ for update by

$$B = A_t \cos(\|f(t)\|) + \frac{f(t)}{\|f(t)\|} \sin(\|f(t)\|).$$

6. Calculate $F(B)$, $F(A_t)$, and the increment $dF = F(B) - F(A_t)$.
7. Accept B with probability $\min\{e^{dF/T_t}, 1\}$. If B is accepted, set $A_{t+1} = B$. Else, set $A_{t+1} = A_t$.
8. If $t < N$, set $T_{t+1} = T_t/\gamma$ and $t = t + 1$, and go to Step 3. Else, let $\hat{A} = A_t$ and stop.

5.3 An Alternative Interpretation of OFA

Unlike linear dimension-reduction methods that rely only on orthogonal projection onto a subspace of the original feature space, OFA allows general linear mappings to a k-dimensional feature space. In this section, we show that if we are willing to consider metrics other than the Euclidean metric, then dimension reduction and subsequent data classification with OFA may be viewed as obtained from an orthogonal projection onto a subspace of the original feature space. If A is a rank r matrix, take a singular value decomposition

$$A = U \Sigma V^T, \tag{13}$$

where U and V are orthogonal matrices of dimensions k and m, respectively, and Σ is a $k \times m$ matrix whose $r \times r$ northwest quadrant is diagonal with positive eigenvalues and whose remaining entries are all zero. Let H be the r-dimensional subspace of $\mathbb{R}^m$ spanned by the first r columns of V and denote the orthogonal projection of a vector $x \in \mathbb{R}^m$ onto H by x_H. Then,

$$Ax \cdot Ay = y^T (A^T A) x = y^T K x = y_H^T K x_H, \tag{14}$$

for any $x, y \in \mathbb{R}^m$, where $K = A^T A$ is a positive semi-definite symmetric matrix. In particular,

$$\|Ax - Ay\|^2 = (x_H - y_H)^T K (x_H - y_H). \tag{15}$$

This means that the Euclidean distance between feature vectors in the reduced space $\mathbb{R}^k$ can be interpreted as the distance between the projected vectors x_H and y_H in the original feature space with respect to the new metric

$$d(x_H, y_H) = \sqrt{(x_H - y_H)^T K (x_H - y_H)}\,. \tag{16}$$

Note that the subspace H is spanned by the eigenvectors of K associated with its non-zero eigenvalues, so that (16) does define a metric on H. Thus, OFA may be viewed as a technique to learn a subspace H of $\mathbb{R}^m$ for orthogonal dimension reduction and a positive definite quadratic form on H that are optimal for categorization based on the nearest-neighbor classifier.

6 Image Categorization

We report the results of several image categorization experiments with the Corel-1000 data set described in Section 3. In each experiment, we placed an equal number of images from each class in the training set and used the remaining ones as query images to be indexed by the nearest-neighbor classifier applied to a reduced feature learned with OFA. Initially, an image is represented by an SH-feature vector $h(I, \mathcal{F})$ of dimension 165 obtained from the 11-bin histograms associated with 5 filters applied to the R, G, and B channels. OFA was used to reduce the dimension to $k = 9$. Table 1 shows the categorization performance: T denotes the total number of images in the training set and categorization performance refers to the rate of correct indexing using all $1,000 - T$ images outside the training set as queries.

Table 1. Results of categorization experiments with the Corel-1000 data set. T is the number of training images and the dimension of the reduced feature space is 9.

T	Categorization Performance
600	85.5%
400	84.5%
200	71.7%

7 Image Retrieval

We now use the reduced features learned with OFA to retrieve images from the database. We begin with the remark that the reduced representation was optimized to categorize query images with the nearest neighbor classifier, but not necessarily to rank matches to a query image correctly according to distances in feature space. Thus, in contrast with the retrieval strategy based solely on distances adopted, for example, in [8] and [2], we propose to exploit the strengths of the image categorization method in a more essential way.

Let $A\colon \mathbb{R}^m \to \mathbb{R}^k$ be the optimal linear dimension-reduction map learned with OFA. If I is an image and $h(I, \mathcal{F}) \in \mathbb{R}^m$ is the associated SH-feature vector, we let x denote its projection to $\mathbb{R}^k$; that is,

$$x = Ah(I, \mathcal{F}). \tag{17}$$

If there are P classes of images, for each $1 \leqslant i \leqslant P$, let x_i be the reduced feature vector of the training image in class i closest to x and let

$$d_i(I) = \|x - x_i\| \tag{18}$$

be the distance from I to class i in reduced feature space.

Given a query image I and a positive integer ℓ, the goal is to retrieve a ranked list of ℓ images from the database. We assume that all images in the database have been indexed using the representation learned with OFA. Given I, rank the classes according to increasing values of the distances $d_i(I)$. We retrieve images as follows: select as many images as possible from the first class; once that class is exhausted, we proceed to the second and iterate the procedure until ℓ images are obtained. Within each class, the images are retrieved and ranked according to their Euclidean distances to I as measured in the reduced feature space.

7.1 Experimental Results

We report the results of retrieval experiments with the Corel-1000 dataset. To make objective comparisons with other systems, we only use query images that are part of the database. Since each class contains 100 images, the maximum possible number of matches to a query image is 100, where a match is an image that belongs to the same class. We first compare retrieval results using OFA learning with those obtained with SIMPLIcity and spectral histograms, as described in Section 3. We calculated the mean values $\bar{p}_i$ and $\bar{r}_I$ of the weighted precision and rank as defined in (4). The plots shown in Figure 4 show a significant improvement in retrieval performance with a learning component. OFA was used with 400 training images (OFA-400).

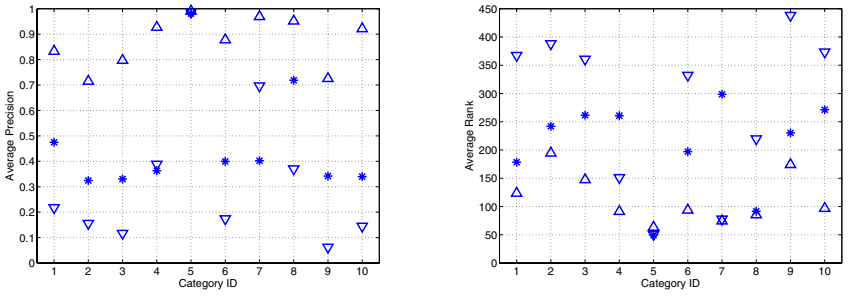


Fig. 4. (a) Plots of $\bar{p}_i$ and $\bar{r}_i$, $1 \leqslant i \leqslant 10$. The methods are labeled as: ($\triangledown$) spectral histogram; ($*$) SIMPLIcity; ($\triangle$) OFA-400.

We further quantify retrieval performance, as follows. For an image I and a positive integer ℓ, let m_ℓ be the number of matching images among the top ℓ returns. Let

$$p_\ell(I) = \frac{m_\ell(I)}{\ell} \quad \text{and} \quad r_\ell(I) = \frac{m_\ell(I)}{100} \tag{19}$$

be the precision and recall rates for ℓ returns for image I. The average precision and average recall for the top ℓ returns are defined as

$$p_\ell = \frac{\sum_I p_\ell(I)}{1000} \quad \text{and} \quad r_\ell = \frac{\sum_I r_\ell(I)}{1000}\,, \tag{20}$$

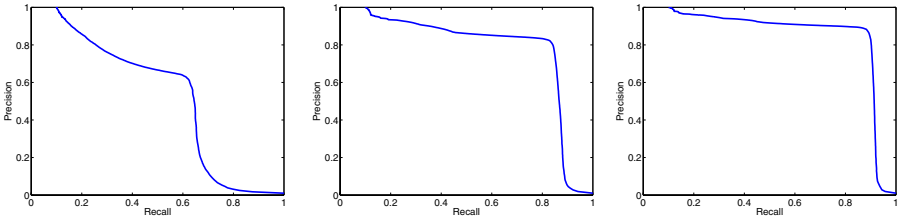


Fig. 5. Corel-1000: average-precision $\times$ average-recall plots for 200, 400 and 600 training images

Table 2. Retrieval results for OFA with T training images. Average retrieval precision (p_ℓ) and average recall (r_ℓ) for the top ℓ matches.

$T = 600$

ℓ	10	20	40	70	100	200	500
p_ℓ	0.925	0.921	0.916	0.909	0.883	0.461	0.192
r_ℓ	0.093	0.184	0.366	0.636	0.883	0.922	0.962

$T = 400$

ℓ	10	20	40	70	100	200	500
p_ℓ	0.889	0.882	0.875	0.861	0.825	0.437	0.188
r_ℓ	0.089	0.176	0.345	0.603	0.825	0.876	0.938

$T = 200$

ℓ	10	20	40	70	100	200	500
p_ℓ	0.731	0.690	0.660	0.649	0.620	0.361	0.176
r_ℓ	0.073	0.138	0.264	0.454	0.620	0.722	0.881

respectively. Here, the sum is taken over all 1,000 images in the database. Note that, for a perfect retrieval system, $p_\ell = 1$, for $1 \leqslant \ell \leqslant 100$, gradually decaying to $p_{1000} = 0.1$ as ℓ increases. Similarly, $r_\ell = 1$, for $\ell \geq 100$, and decays with ℓ to $r_1 = 0.01$.

Table 2 shows several values of the average precision and the average recall based on a 9-dimensional representation learned with T training images. The full average-precision $\times$ average-recall plots are shown in Figure 5. Figure 6 shows the top 10 returns for a few images in the database in an experiment with 400 training images. In each group, the first image is the query image, which is also the top return.

8 Summary and Discussion

We employed a representation of images by the histograms of their spectral components for content-based image categorization and retrieval. A feature learning technique, referred to as Optimal Factor Analysis, was developed to reduce the dimension of the representation and optimize the discriminative ability of the nearest-neighbor classifier. Several experiments were carried out and the results demonstrate a significant improvement in retrieval performance over a number of existing retrieval systems. Refinements

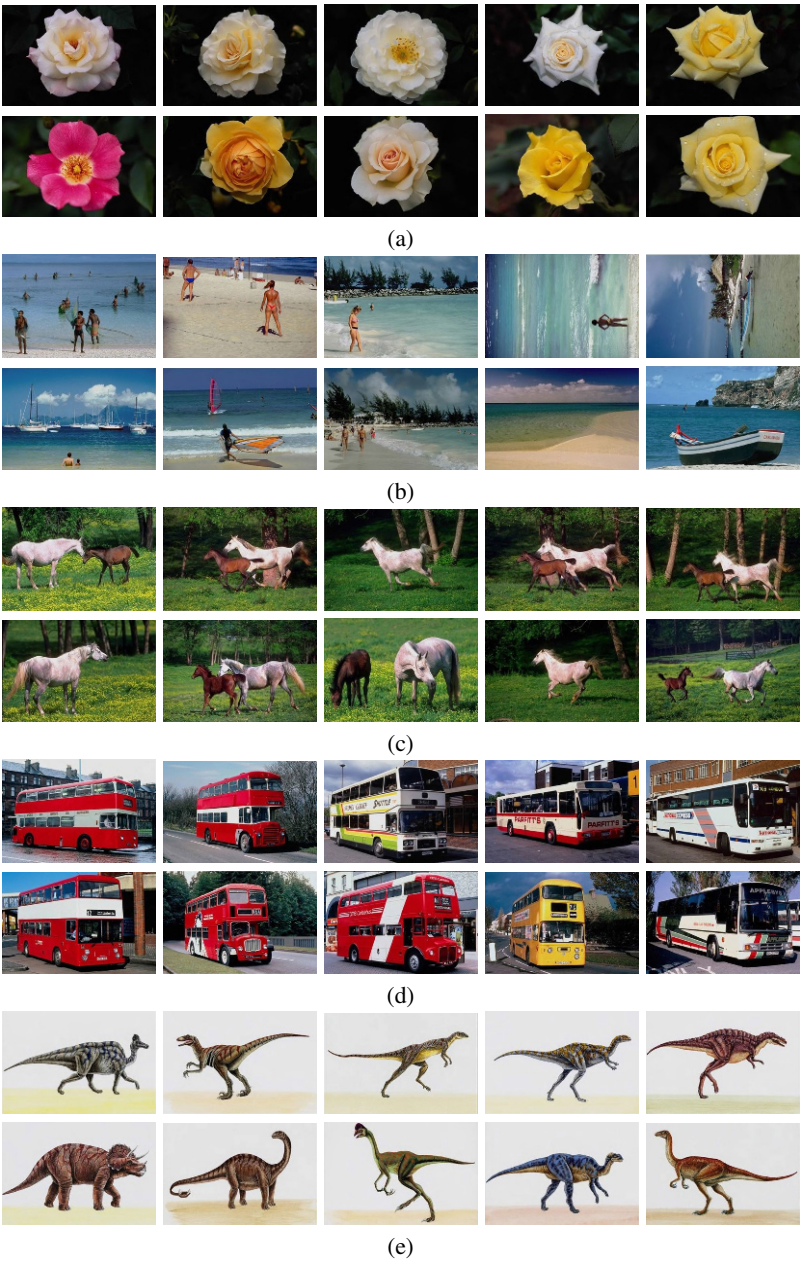


Fig. 6. Examples of top-10 returns. In each group, the first image is the query, which is also the top return.

of the methods will be investigated in future work to obtain sparse representations and to incorporate kernel techniques to cope with nonlinearity in data geometry. Computational strategies for faster retrieval as well as a modified version of the OFA cost

function that allows a more efficient estimation of the gradient also will be investigated in future work.

Acknowledgements. This work was supported in part by NSF grants CCF-0514743 and IIS-0307998.

References

1. Carson, C., Thomas, M., Belongie, S., Hellerstein, J., Malik, J.: Blobworld: a system for region-based image indexing and retrieval. In: Proc. Visual Information Systems, pp. 509–516 (1999)
2. Hoi, S., Liu, W., Lyu, M., Ma, W.-Y.: Learning distance metrics with contextual constraints for image retrieval. In: Proc. CVPR 2006 (2006)
3. Liu, X., Mio, W.: Splitting factor analysis and multi-class boosting. In: Proc. ICIP 2006 (2006)
4. Liu, X., Srivastava, A., Gallivan, K.: Optimal linear representations of images for object recognition. IEEE Trans. Pattern Analysis and Machine Intelligence 26, 662–666 (2004)
5. Portilla, J., Simoncelli, E.: A parametric texture model based on joint statistics of complex wavelet coeficients. International Journal of Computer Vision 40, 49–70 (2000)
6. Rubner, Y., Guibas, L., Tomasi, C.: The earth mover's distance, multi-dimensional scaling, and color-based image databases. In: Proc. DARPA Image Understanding Workshop, pp. 661–668 (1997)
7. Smith, J., Li, C.: Image classification and querying using composite region templates. Computer Vision and Image Understanding 75(9), 165–174 (1999)
8. Wang, J., Li, J., Wiederhold, G.: SIMPLIcity: Semantics-sensitive integrated matching for picture libraries. IEEE Trans. on Pattern Analysis and Machine Intelligence 23(9), 947–963 (2001)
9. Wu, Y., Zhu, S., Liu, X.: Equivalence of Julesz ensembles and FRAME models. International Journal of Computer Vision 38, 247–265 (2000)
10. Yin, P.-Y., Bhanu, B., Chang, K.-C., Dong, A.: Integrating relevance feedback techniques for image retrieval using reinforcement learning. IEEE Trans. on Pattern Analysis and Machine Intelligence 27(10), 1536–1551 (2005)
11. Zhu, S., Wu, Y., Mumford, D.: Filters, random fields and maximum entropy (FRAME). International Journal of Computer Vision 27, 1–20 (1998)

Biased Manifold Embedding: Supervised Isomap for Person-Independent Head Pose Estimation

Vineeth Balasubramanian and Sethuraman Panchanathan

Center for Cognitive Ubiquitous Computing (CUbiC)
Arizona State University, U.S.A.
vineeth.nb@asu.edu, panch@asu.edu

Abstract. An integral component of face processing research is estimation of head orientation from face images. Head pose estimation bears importance in several applications in biometrics, human-computer interfaces, driver monitoring systems, video conferencing and social interaction enhancement programs. A recent trend in head pose estimation research has been the use of manifold learning techniques to capture the underlying geometry of the images. Face images with varying pose angles can be considered to be lying on a smooth low-dimensional manifold in high-dimensional image feature space. However, with real-world images, manifold learning techniques often fail because of their reliance on a geometric structure, which is often distorted due to noise, illumination changes and other variations. Also, when there are face images of multiple individuals with varying pose angles, manifold learning techniques often do not give accurate results. In this work, we introduce the formulation of a novel framework for supervised manifold learning called Biased Manifold Embedding to obtain improved performance in person-independent head pose estimation. While this framework goes beyond pose estimation, and can be applied to all regression applications, this work is focused on formulating the framework and validating its performace using the Isomap technique for head pose estimation. The work was carried out on face images from the FacePix database, which contains 181 face images each of 30 individuals with pose angle variations at a granularity of 1 $^{\circ}$. A Generalized Regression Neural Network (GRNN) was used to learn the non-linear mapping, and linear multi-variate regression was adopted on the low-dimensional space to obtain the pose angle. Results showed that the approach holds promise, with estimation errors substantially lower than similar efforts in the past using manifold learning techniques for head pose estimation.

Keywords: Head pose estimation, Manifold learning, Supervised learning, Face processing.

1 Introduction

As human-centered computing applications grow each day, human face analysis has grown in its importance as a problem studied by several research communities. The estimation of head pose angle from face images is a significant sub-problem in several applications like 3D face modeling, gaze direction detection, driver monitoring safety systems, etc. Further, realistic solutions to the problem of face recognition have to be

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 177–188, 2008.

able to handle significant head pose variations, thereby leading to the gain in importance of the automatic estimation of the orientation of the head relative to the camera-centered co-ordinate system. While coarse head pose estimation has been successful to a large extent [2], accurate person-independent pose estimation, which is very crucial for applications like 3D face modeling, is still being worked on.

Current literature [4], [8], [11] separates the existing methods for head pose estimation into distinct categories:

- *Shape-based geometric analysis*, where head pose is discerned from geometric information like the configuration of facial landmarks.
- *Model-based methods*, where non-linear parametric models are derived before using a classifier like a neural network (Eg. Active Appearance Models (AAMs)).
- *Appearance-based methods*, where the pose estimation problem is viewed as a pattern classification problem on image feature spaces.
- *Template matching approaches*, which are largely based on nearest neighbor classification against texture templates/signatures.
- *Dimensionality reduction based approaches*, where linear/non-linear embedding of the face images is used for pose estimation.

To overcome data redundancy and obtain compact representations of face images, earlier work [3], [8], [4] suggests to consider the high-dimensional face image data as a set of geometrically related points lying on a smooth manifold in the high-dimensional feature space.

Different poses of the head, although captured in high-dimensional image feature spaces, can be visualized as data points lying on a low-dimensional manifold in the high-dimensional space. Raytchev et al [8] stated that the dimension of this manifold is equivalent to the number of degrees of freedom in the movement during data capture. For example, images of the human face with different angles of pose rotation (yaw, tilt and roll) can intrinsically be conceptualized as a 3D manifold in image feature space. This conceptualization resulted in a host of dimensionality reduction techniques that are based on the relative geometry of the data points in high-dimensional space. This is the idea that underlies the family of non-linear dimensionality reduction techniques under the umbrella of manifold learning, like Isomap, Locally Linear Embedding (LLE), Laplacian Eigenmaps, Local Tangent Space Alignment (LTSA), etc, which have become popular in recent times.

In prior work in this domain, [8] and [5] employed a straight-forward approach to learn the non-linear mapping onto the low-dimensional space through manifold learning, and estimated the pose angle using a pose parameter map. In the work carried out so far, the pose information of the given face images is ignored while computing the embedding. In this light, we propose a novel improvement to traditional manifold learning techniques, called the Biased Manifold Embedding approach, which provides a bias to the manifold-based embedding process, using pose information from the given face image data. While the proposed Biased Manifold Embedding method is illustrated using Isomap in this paper, it can easily be extended to other manifold learning techniques with minor adaptations. As broader impact, the work proposed here is a framework for a supervised approach to manifold-based non-linear dimensionality reduction techniques across all regression problems.

We discuss the background with a brief description of the Isomap algorithm, followed by related work and an insight into the significance of our work in Section 2. Section 3 details the mathematical formulation of the proposed Biased Manifold Embedding method. The experimental setup and the methodology of our experiments are briefed in Section 4. The results of the experiments are discussed in Section 5. We then discuss the advantages and limitations of the approach in the concluding section in Section 6, and provide future directions to this work.

2 Background

2.1 Non-linear Dimensionality Reduction Using Isomap

Finding low-dimensional representations of high-dimensional data is a common problem in science and engineering. High-dimensional observations are prevalent in all fields: images, spectral data, instrument readings, etc. Techniques like Principal Component Analysis (PCA) are recognized as linear dimensionality reduction techniques, because of the linear projection matrix obtained from the eigen vectors of the covariance matrix, while techniques like Multi-Dimensional Scaling (MDS) are grouped under non-linear dimensionality reduction techniques. However, MDS uses the L2 (Euclidean) distance between data points in the high-dimensional space to capture their similarities. If the data points were to lie on a manifold in the high-dimensional space, Euclidean distances do not capture the geometric relationship between the data points. In such cases, it is beneficial to consider the geodesic (along the surface on which the data points lie) distances between the data points to obtain a more truthful representation of the data.

To capture the global geometry of the data points, Tanenbaum et al [10] proposed Isomap to compute an isometric low-dimensional embedding of a given set of high-dimensional data points (See Algorithm 1).

While Isomap captures the global geometry of the data points in the high-dimensional space, the disadvantage of this family of manifold learning techniques is the lack of a projection matrix to embed out-of-sample data points after the training phase. This makes the method more suited for data visualization, rather than classification problems. However, these techniques capture the relative geometry of data points,and this enthuses researchers to adopt this methodology to solve problems like head pose estimation, where the data is known to possess geometric relationships in a high-dimensional space. Figure 1 shows the visualization results of using Isomap to embed face images onto 2 dimensions. Faces of 10 individuals with 11 pose angles (-75° to +75° in increments of 15) were used to perform this embedding. The feature space considered here was the space of grayscale pixel intensities. As evident from this figure, the embedding of the face images reflects an intrinsic ordering on the corresponding pose angles. While this indicates the sensitivity of this approach to face images with varying pose angles, the clutter of images on the trajectory suggests that fine estimation of pose angle still remains a challenging problem.

Algorithm 1. Isomap algorithm.

Step 1: Construct Neighborhood Graph

Determine the neighbors of a point on the manifold $\mathcal{M}$. The neighbors are identified as the data points within a e-radius of a given point, or one among the k nearest neighbors in terms of Euclidean distance from the given point. The neighborhood of each point is represented as a weighted graph $\mathcal{G}$ over the data points, with each edge characterized by the distance $d_x(i,j)$ between the pair of neighboring points.

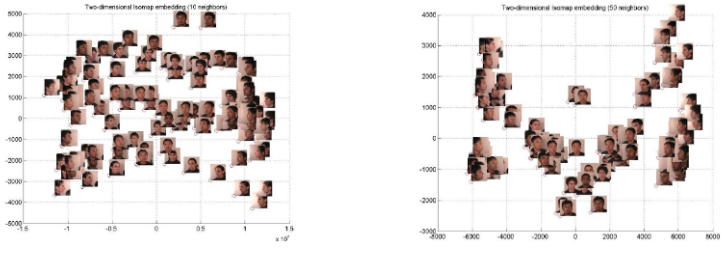
Step 2: Compute Shortest Paths

Estimate the geodesic distances $d_M(i,j)$ between all pairs of points on the manifold $\mathcal{M}$ by computing their shortest path distance in the graph $\mathcal{G}$. This is done using the Floyd's or Djkstraa's algorithm. For example:

$$d_M(x_i,x_j) = mind_M(x_i,x_j), d_M(x_i,x_k) + d_M(x_k,x_j)$$

Step 3: Derive Low-dimensional Embedding

Apply classical MDS to the geodesic distances matrix $D_M = d_M(i,j)$, deriving an embedding of the data in a low-dimensional Euclidean space $\mathbf{Y}$ that best preserves the estimated intrinsic geometry of the manifold.



(a) Isomap embedding with 10 neighbors. (b) Isomap embedding with 50 neighbors.

Fig. 1. Embedding of face images with varying poses onto 2 dimensions using Isomap with different neighbor parameters

2.2 Related Work

Over the last few years since the arrival of manifold learning techniques, a reasonable amount of work has been done using manifold-based dimensionality reduction techniques for head pose estimation. Chen et al [3] considered multi-view face images as lying on a manifold in high-dimensional feature space. However, they compared the effectiveness of Kernel Discriminant Analysis against Support Vector Machines in learning the manifold gradient direction in the high-dimensional feature space, and did not adopt manifold learning for non-linear dimensionality reduction. Raytchev et al [8] studied the effectiveness of Isomap for head pose estimation against other view representation approaches like the Linear Subspace model and Locality Preserving Projections (LPP). While their work established the possible gain in accuracy through use of manifold learning techniques, the face images used by them were sampled at pose

angle increments of 15°, and relied on the robustness of the captured mapping and interpolation to obtain the precise pose angle estimate. Hu et al [5] developed a unified embedding approach for multiple individuals, where the embedding obtained from Isomap for a single individual was parametrically modeled as an ellipse. The ellipses for different individuals were subsequently normalized through scale, translation and rotation based transformations to obtain a unified embedding. In more recent work, Fu and Huang [4] presented an appearance-based strategy for head pose estimation using a supervised form of Graph Embedding, which internally used the idea of Locally Linear Embedding (LLE). This work mainly focussed on obtaining a linearization of manifold learning techniques to treat out-of-sample data points.

There has been recent work by [9] and [12] to obtain a supervised approach to manifold learning techniques. However, their approaches are strictly oriented towards classification problems, and do not exploit the label information as possible for regression problems like head pose estimation.

2.3 Proposed Approach

While manifold learning techniques like Isomap capture the global geometrical relationship between data points in the high-dimensional image feature space, they do not use the pose label information of the training data samples. Unlike class labels in classification problems, pose information can be viewed as an ordered single-dimensional label with an established distance metric. This can provide valuable input to the embedding process.

In this work, we propose a biased manifold-based embedding for head pose estimation. We use the given pose information to bias the non-linear embedding to obtain accurate pose angle estimation. The significance of our contribution is realized in the fact that the proposed Biased Manifold Embedding method, although validated in this work with Isomap, can be extended to other manifold learning techniques with minor modifications, and in general, can be applied to all regression problems that use manifold learning methods. In addition, while most current approaches use face images sampled with pose angles at increments of 10-15° [8], we use the FacePix database [7] that includes images of faces taken at a wide range of precisely measured pose angles with a readily available granularity of 1°. This reinforces the validity of our experiments with the proposed approach.

3 Biased Manifold Embedding

In the Biased Manifold Embedding method, we propose to use the pose angle information of the training data samples to obtain a more meaningful embedding with a view to solve the problem of pose estimation. The fundamental idea of our approach is that face images with nearer pose angles must be nearer to each other in the low-dimensional embedding, and images with farther pose angles are placed farther, irrespective of the identity of the individual. We achieve this with a modification to the computation of the geodesic distance matrix. Since a distance metric can easily be defined on the pose angle values, the problem of finding closeness of pose angles is straight-forward.

The mathematical formulation of the Biased Manifold Embedding method is given below. We would like the ideal modified geodesic distance between a pair of data points to be of the form:

$$\tilde{D}(i,j) = f(P(i,j)) \otimes D(i,j)$$

where $D(i,j)$ (= d_M in Algorithm 1) is the geodesic distance between two data points x_i and x_j, $\tilde{D}(i,j)$ is the modified biased geodesic distance, $P(i,j)$ is the pose distance between x_i and x_j, f is any function of the pose distance, and $\otimes$ is a binary operator. If $\otimes$ was chosen as the multiplication operation, the function f would be chosen as inversely proportional to the pose distance, $P(i,j)$. In a more general perspective, the function f could be picked from the family of reciprocal functions ($f \in \mathcal{F}_R$) based on the needs of an application. In this work, we choose the function as:

$$f(P(i,j)) = \frac{1}{max_{m,n}P(m,n) - P(i,j)}$$

This function could be replaced by an inverse exponential or quadratic function of the pose distance. In order to ensure that the biased geodesic distance values are well-separated for different pose distances, we multiply this quantity by a function of the pose distance:

$$\tilde{D}(i,j) = \frac{\alpha(P(i,j))}{max_{m,n}P(m,n) - P(i,j)} * D(i,j)$$

where the function a is directly proportional to the pose distance, P(i, j), and is defined in our work as:

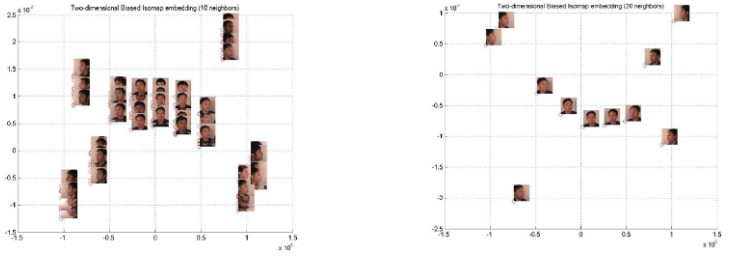
$$\alpha(P(i,j)) = \beta * |P(i,j)|$$

where β is a constant of proportionality, and allows parametric variation for performance tuning. In our work, we have used the pose distance as the one-dimensional distance i.e. $P(i,j) = |Pi - Pj|$, where P_k is the pose angle of x_k. In summary, the biased geodesic distance between a pair of points can be given by:

$$\tilde{D}(i,j) = \begin{cases} \frac{\alpha(P(i,j))}{max_{m,n}P(m,n) - P(i,j)} * D(i,j) & P(i,j) \neq 0, \\ 0 & P(i,j) = 0. \end{cases} \quad (1)$$

Classical MDS is applied on this biased geodesic distance matrix to obtain the embedding. The proposed modification impacts only the computation of the geodesic distance matrix, and hence, can easily be extended to other manifold-based dimensionality reduction techniques that use the geodesic distance.

Figure 2 shows the results of using Biased Isomap to embed the same facial images used in Figure 1 onto 2 dimensions. The embedded images establish the tendency of the method to elicit person-independent representations of the pose angles of the given facial images. As expected from the formulation of the method (see Figure 2), the face images of all individuals with the same pose angle have merged onto the same data point in 2 dimensions. This renders an embedding that is more conducive to determine the pose angle from the face images.



(a) Biased Isomap embedding with 10 neighbors

(b) Biased Isomap embedding with 20 neighbors

Fig. 2. Biased Isomap Embedding of face images with varying poses onto 2 dimensions. Note in 2(b) that all the face images with the same pose angle have merged onto the same 2D point.

4 Experimental Setup and Methodology

The proposed Biased Isomap Embedding approach was compared against the traditional Isomap method for non-linear dimensionality reduction in the head pose angle estimation process. We used the FacePix face database [7] (see Figure 3) built at the Center for Cognitive Ubiquitous Computing (CUbiC), which has face images with precisely measured pose variation. In this work, we consider a set of 2184 face images, consisting of 24 individuals with pose angles varying from -90° to +90° in increments of 2°. The images were subsampled to 32 x 32 resolution, and different feature spaces of the images were considered for the experiments. The results presented here include the grayscale pixel intensity feature space and the Laplacian of Gaussian (LoG) transformed image feature space (see Figure 4). The LoG transform was used since pose variation in face images is a result of geometric transformation, and texture information may not be really useful for the pose estimation problem. This was also reflected in preliminary experiments conducted with Gabor filters and Fourier-Mellin transformed images. The images were subsequently rasterized and normalized.

Non-linear dimensionality reduction techniques like manifold learning do not provide a projection matrix to handle test data points. While different approaches have been used by earlier researchers to capture the mapping from the high-dimensional feature space to the low-dimensional embedding, we adopted a Generalized Regression Neural Network (GRNN) with Radial Basis Functions to learn the non-linear mapping. This approach has been adopted earlier by Zhao et al [13]. Additionally, the parameters involved in training the network (just the spread of the Radial Basis Function) are minimal, thereby facilitating better evaluation of the proposed method. Once the low-dimensional embedding was obtained, linear multi-variate regression was used to obtain the pose angle of the test image.

The proposed Biased Isomap Embedding method was compared with the traditional Isomap approach using resubstitution and 8-fold cross-validation models. In the resubstitution model, 100 data points were randomly chosen from the training sample for the testing phase. The error in estimation of the pose angle was used as the metric for

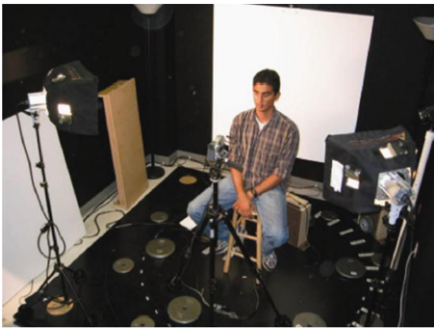


Fig. 3. The data capture setup for FacePix

(a) Grayscale image

(b) Laplacian of Gaussian (LoG) tranformed image

Fig. 4. Image feature spaces used for the experiments

performance evaluation. In the 8-fold cross-validation model, face images of 3 individuals were used for the testing phase in each fold, while all the remaining images were used in the training phase. In addition to these experiments, the variation in accuracy of the proposed method with the embedding dimension and the number of neighbors for the embedding was studied.

5 Results and Discussion

The results for the resubstitution model are presented in Table 1. The improved performance of the Biased Isomap Embedding method for head pose estimation is unanimously reflected in the significant reduction in error values across the image feature spaces. However, validation using the resubstitution model is preliminary since test samples are picked from the training sample set itself. For more robust validation, we implemented 8-fold cross-validation over the images from 24 individuals. The results of these experiments are shown in Table 2. The results with the cross-validation model corroborate our claim of the performance gain. Both of these experiments were carried out with an embedding dimension of 8, with a choice of 50 neighbors for the embedding. The pose angle estimate error is consistently under 4°, which is a substantial improvement over earlier work [8].

Table 1. Results using the resubstitution model

Feature Space	Error using traditional Isomap	Error using Biased Isomap
Grayscale	11.39	1.98
Laplacian of Gaussian	8.80	2.31

Table 2. Results using the 8-fold cross-validation model

Feature Space	Error using traditional Isomap	Error using Biased Isomap
Grayscale	10.55	3.68
Laplacian of Gaussian	9.10	3.38

Table 3. Analysis of performance with varying dimensions of embedding

Dimension of Embedding	Error using traditional Isomap	Error using Biased Isomap
100	10.41	5.02
50	10.86	5.04
20	11.35	5.04
8	12.96	5.07
5	12.57	5.05
3	16.21	5.66

In addition, the performance of the Biased Manifold Embedding was analyzed with varying dimensions of embedding, and choice of the number of neighbors used for embedding. Table 3 captures the results for different embedding dimensions with the number of neighbors fixed at 50. Table 4 captures the results for varying number of neighbors for the embedding with the embedding dimension fixed at 8. Grayscale pixel intensities of the face images were used for these independent experiments.

As evident from the results, the significant reduction in the error of estimation of pose angle substantiates the effectivness of the proposed approach. In addition, as the results in Tables 2, 3 and 4 illustrate, the Biased Manifold Embedding method is robust to variations in feature spaces, dimensions of embedding and choice of number of neighbors. While the traditional Isomap embedding has fluctuating results for these parameters, the range of error values obtained for the Biased Manifold Embedding method across these parameter changes suggests the high stability of the method, thanks to the biasing of the embedding.

A summary of results from related approaches is presented in Table 5. Note that these results are extracted from earlier work which have had different experimental

Table 4. Analysis of performance with varying number of neighbors for embedding

Number of Neighbors	Error using traditional Isomap	Error using Biased Isomap
30	11.56	5.10
50	12.96	5.06
100	13.83	5.03
200	12.59	5.06
500	14.36	5.07

Table 5. Summary of head pose estimation results from related approaches in recent years

Reference	Method	Best result: Error/Accuracy	Notes
[3]	Fisher Manifold Learning	About 3 °	Face images only in $[-10\,^\circ, 10\,^\circ]$ interval
[6]	Kernel PCA + Support Vector Machines	97%	Face images only in 10 ° intervals. This was framed as a classification problem of identifying the pose angle as one of these intervals.
[8]	Isomap	About 11 °	Face images sampled at 15 ° increments
[8]	LPP	About 15 °	Face images sampled at 15 ° increments
[4]	LEA	About 2 °	Best results so far

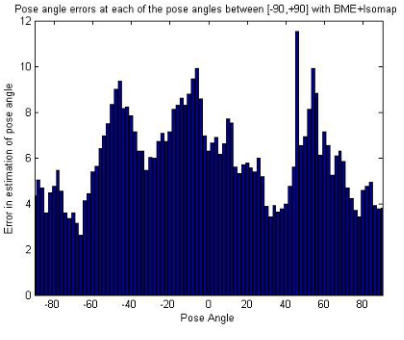


Fig. 5. Analysis of the average error in pose estimation for each of the views between $[-90\,^\circ, +90\,^\circ]$

design criteria, and may not be compared directly. This table has been presented just to provide an idea of the results have been obtained so far.

For a better understanding of the results, we analyzed how the errors in the pose estimation process were spread out on the interval $[-90\,^\circ, +90\,^\circ]$. Figure 5 shows the

head pose estimation error in each of the views in this pose angle interval. While we expected to see a better performance at the frontal view, this was not very evident in any of the three approaches. We also hoped to identify particular regions of pose angle views of face images where the framework consistently performs relatively poor. However, these plots do not provide any coherent information on identifying such views of face images.

6 Conclusions

We have proposed the Biased Manifold Embedding method, a novel supervised approach to manifold learning techniques for regression problems. The proposed method was validated for accurate person-independent head pose estimation. The use of pose information in the manifold embedding process improved the performance of the pose estimation process significantly. The pose angle estimates obtained using this method are accurate, and can be relied upon with an error margin of 3-4°. Our experiments also demonstrated that the method is robust to variations in feature spaces, dimensionality of embedding and the choice of the number of neighbors for the embedding. The proposed method can easily be extended from the current Isomap implementation to cover the envelop of other manifold learning techniques, and can be developed as a framework for biased manifold learning to cater to all regression problems at large.

6.1 Limitations and Future Work

As mentioned earlier, a significant drawback of manifold learning techniques is the lack of a projection matrix to treat new data points. While we used the GRNN to learn the non-linear mapping in this work, there have been other approaches adopted by various researchers. Bengio et al [1] proposed a mathematical formulation focussed to overcome this problem. We plan to use these approaches to support the validity of our approach. Besides, we intend to extend the Biased Manifold Embedding implementation to LLE and Laplacian Eigenmaps to establish it as a framework for non-linear dimensionality reduction in regression applications. On a lesser significant note, another limitation of the current approach is that the number of neighbors chosen to obtain the embeddding has to be more than the number of individuals in the face images. This is because different individuals with the same pose angle are assigned a zero distance value in the biased geodesic distance matrix. We plan to modify our algorithm to overcome this limitation. In addition, the function of pose distance used to bias the geodesic distance matrix can be varied to study the applicability of different reciprocal functions for pose estimation.

References

1. Bengio, Y., Paiement, J.F., Vincent, P., Delalleau, O.: Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering (2004)
2. Brown, L., Tian, Y.L.: Comparative study of coarse head pose estimation (2002)

3. Chen, L., Zhang, L., Hu, Y.X., Li, M.J., Zhang, H.J.: Head pose estimation using fisher manifold learning. In: IEEE International Workshop on Analysis and Modeling of Faces and Gestures (AMFG 2003), Nice, France, pp. 203–207 (2003)
4. Fu, Y., Huang, T.S.: Graph embedded analysis for head pose estimation. In: 7th International Conference on Automatic Face and Gesture Recognition, Southampton, UK (2006)
5. Hu, N., Huang, W., Ranganath, S.: Head pose estimation by non-linear embedding and mapping. In: IEEE International Conference on Image Processing, Genova, Genova, pp. 342–345 (2005)
6. Li, S., Fu, Q., Gu, L., Scholkopf, B., Cheng, Y., Zhang, H.: Kernel machine based learning for multi-view face detection and pose estimation. In: IEEE International Conference on Computer Vision (ICCV 2001), vol. 2, pp. 674–679 (2001)
7. Little, G., Krishna, S., Black, J., Panchanathan, S.: A methodology for evaluating robustness of face recognition algorithms with respect to variations in pose and illumination angle. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, Philadelphia, USA, pp. 89–92 (2005)
8. Raytchev, B., Yoda, I., Sakaue, K.: Head pose estimation by nonlinear manifold learning. In: 17th International Conference on Pattern Recognition (ICPR 2004), Cambridge, UK (2004)
9. Ridder, D.d., Kouropteva, O., Okun, O., Pietikainen, M., Duin, R.P.: Supervised locally linear embedding. In: International Conference on Artificial Neural Networks and Neural Information Processing, pp. 333–341 (2003)
10. Tanenbaum, J.B., Silva, V.d., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290(5500), 2319–2323 (2000)
11. Wenzel, M.T., Schiffmann, W.H.: Head pose estimation of partially occluded faces. In: Second Canadian Conference on Computer and Robot Vision (CRV 2005), Victoria, Canada, pp. 353–360 (2005)
12. Yu, J., Tian, Q.: Learning image manifolds by semantic subspace projection. In: ACM Multimedia, Santa Barbara, CA, USA (2006)
13. Zhao, Q., Zhang, D., Lu, H.: Supervised lle in ica space for facial expression recognition. In: International Conference on Neural Networks and Brain (ICNNB 2005), Beijing, China, vol. 3, pp. 1970–1975 (2005)

Part IV
Motion, Tracking and Stereo Vision

High Performance Model-Based Object Detection and Tracking

Alexander Ladikos, Selim Benhimane, and Nassir Navab

Technische Universität München, Boltzmannstr. 3, 85748 Garching, Germany
ladikos@in.tum.de

Abstract. We present a complete real-time model-based tracking system for piecewise-planar objects which combines template-based and feature-based approaches. Our contributions are an extension to the ESM algorithm used for template-based tracking and the formulation of a feature-based tracking approach, which is specifically tailored for use in a real-time setting. In order to cope with highly dynamic scenarios, such as illumination changes, partial occlusions and fast object movement, the system adaptively switches between template-based tracking, feature-based tracking and a global initialization phase. Our tracking system achieves real-time performance by applying a coarse-to-fine optimization approach and includes means to detect a loss of track.

Keywords: Real-Time Vision, Model-Based Object Tracking, Feature-based Tracking, Template-based Tracking.

1 Introduction

Tracking lays the foundation for many application areas, including Augmented Reality, visual servoing and vision-based industrial applications. Consequently, there is a huge amount of related publications. The methods used for real-time 3D-tracking can be roughly divided into four categories: Line-based tracking, template-based tracking, feature-based tracking and hybrid approaches.

Line-based tracking requires a line model of the tracked object. The pose is determined by matching a projection of the line model to the lines extracted in the image. One of the first publications in this field was [5]. Recently a real-time line tracking system which uses multiple-hypothesis line tracking was proposed in [16]. The main disadvantage of line tracking is that it has severe problems with background clutter and image blurring so that in practice it cannot be applied in the applications we are targeting.

Template-based tracking fits better into our scenarios. It uses a reference template of the object and tracks it using image differences. This works nicely for well-textured objects and small interframe displacements. One of the first publications on template-based tracking [12] was using the optical flow in order to recover the translations in the image plane of the tracked objects. In order to improve the efficiency of the tracking and to deal with more complex objects and/or camera motions, other approaches were proposed [7,1]. In [2] the authors compare these approaches and show that they all have an equivalent convergence rate and frequency up to a first order approximation

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 191–204, 2008.

with some being more efficient than others. A more recently suggested approach is the Efficient Second Order Minimization (ESM) algorithm [4], whose main contribution consists in finding a parametrization and an algorithm, which allow to achieve second-order convergence at the computational cost and consequently the speed of first-order methods.

Similarly to template-based tracking feature-based approaches also require a well-textured object. They work by extracting salient image regions from a reference image and matching them to another image. Each single point in the reference image is compared with other points belonging in a search region in the other image. The one that gives the best similarity measure score is considered as the corresponding one. A common choice for feature extraction is the Harris corner detector [8]. Features can then be matched using normalized cross correlation (NCC) or some other similarity measure [17]. Two recent feature-matching approaches are SIFT [11] and Randomized Trees [10]. Both perform equally well in terms of accuracy. However, despite a recently proposed optimization of SIFT called SURF [3], SIFT has a lower runtime performance than the Randomized Trees, which exhibit a fast feature matching thanks to an offline learning step. In comparison to template-based methods, feature-based approaches can deal with bigger interframe displacements and can even be used for wide-baseline matching if we consider the whole image as the search region. However, wide-baseline approaches are in general too slow for real-time applications. Therefore they are mostly used for initialization rather than tracking. A full tracking system using only features was proposed in [15]. It relies on registered reference images of the object and performs feature matching between reference image and current image as well as between previous image and current image to estimate the pose of the object. However, the frame rate is not very high because of their complex cost function. Moreover image blurring poses a problem for feature extraction.

Hybrid tracking approaches combine two or more of the aforementioned approaches. Some recent related publications include [14], which combines template-based tracking and line-based tracking. In [15] the authors combine line-based tracking and feature-based tracking. Even though these algorithms perform well, the line-based tracking only improves the results for a few cases and might corrupt the result in the case of background clutter. In [13] the authors use a template-based method for tracking small patches on the object, which are then used for a point-based pose estimation. Since this approach uses a template-based method for tracking it cannot deal with fast object motion.

Our proposed system combines template-based and feature-based tracking approaches. The template-based tracking is used as the default tracking since it handles small interframe displacements, image blur and linear illumination changes well. In our system we adopt an extended version of the ESM algorithm, due to its high convergence rate and accuracy. For larger interframe displacements, which cannot be handled by the template-based algorithm, we use a feature-based approach making use of Harris points and NCC. We decided against using both feature-based and template-based tracking at the same time in a combined cost function, since features do not add any precision for small displacements and for big displacements the gradient direction given by ESM is usually erroneous. A combined approach also increases the

computational burden, which not only slows down the tracker but also increases the interframe displacement. For the (re-)initialization we use Randomized Trees, because of their good runtime performance.

The rest of the paper is structured as follows: Section 2 introduces the theoretical background used in our system and section 3 describes our system design. In section 4 we present some simulations with ground-truth and some real-world experimental results. We conclude with section 5.

2 Theoretical Background

Every (4×4) matrix $\mathbf{T}$ defining a 3D rigid body transformation is an element of the special Euclidean group $\mathbb{SE}(3)$. Moreover the Lie-Algebra $\mathfrak{se}(3)$ is linked to $\mathbb{SE}(3)$ through the exponential map. The base elements of $\mathfrak{se}(3)$ can be chosen as follows:

$$\mathbf{A}_1 = \begin{bmatrix} \mathbf{0} & \mathbf{b}_x \\ \mathbf{0} & 0 \end{bmatrix} \quad \mathbf{A}_4 = \begin{bmatrix} [\mathbf{b}_x]_\times & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix}$$

$$\mathbf{A}_2 = \begin{bmatrix} \mathbf{0} & \mathbf{b}_y \\ \mathbf{0} & 0 \end{bmatrix} \quad \mathbf{A}_5 = \begin{bmatrix} [\mathbf{b}_y]_\times & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix}$$

$$\mathbf{A}_3 = \begin{bmatrix} \mathbf{0} & \mathbf{b}_z \\ \mathbf{0} & 0 \end{bmatrix} \quad \mathbf{A}_6 = \begin{bmatrix} [\mathbf{b}_z]_\times & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix}$$

with $\mathbf{b}_x = [1\ 0\ 0]^\top$, $\mathbf{b}_y = [0\ 1\ 0]^\top$ and $\mathbf{b}_z = [0\ 0\ 1]^\top$. The matrices $\mathbf{A}_1$, $\mathbf{A}_2$, $\mathbf{A}_3$ generate the translations and $\mathbf{A}_4$, $\mathbf{A}_5$, $\mathbf{A}_6$ generate the rotations. Consequently, we can parameterize a transformation matrix:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{SE}(3)$$

where $\mathbf{R}$ is the rotation and $\mathbf{t}$ is the translation, using the parameter vector that consists of the coefficients for each base element. Hence given a coefficient vector $\mathbf{x} = [x_1, x_2, ..., x_6]^\top$ the corresponding transformation matrix $\mathbf{T}$ is obtained as:

$$\mathbf{T}(\mathbf{x}) = \exp(\sum_{i=1}^{6} x_i \mathbf{A}_i) \tag{1}$$

In our system we also make heavy use of the relation between the movement of a plane in 3D and its movement in the image, since we suppose that every object can be considered as piecewise planar. As shown in [9] every plane movement induces a homography. Let the plane be $\boldsymbol{\pi} = [\mathbf{n}\ d]^\top$ with normal $\mathbf{n}$ and distance d from the camera. Then the homography describing the transformation of the imaged plane is given by:

$$\mathbf{H}(\mathbf{T}) = \mathbf{K} \left(\mathbf{R} - \frac{\mathbf{t}\mathbf{n}^\top}{d} \right) \mathbf{K}^{-1} \tag{2}$$

where $\mathbf{K}$ are the intrinsic parameters of the camera. The basic cost function used for template-based tracking is defined as follows: Let $\mathcal{I}^*$ be the reference image and $\mathcal{I}$

Fig. 1. Two of the textured models tested for tracking

the current image. Further let $\mathbf{p}$ be the pixel coordinates of the pixels in the reference image and $\widehat{\mathbf{T}}$ an initial pose estimate for the current image. Our goal is to estimate an incremental pose update $\mathbf{T}(\mathbf{x})$ with $\mathbf{x}$ the parameter vector encoding rotation and translation. Let $\mathbf{w}$ be the warping function. The cost function is then given as:

$$f(\mathbf{x}) = \sum_{\mathbf{p}} \left[\mathcal{I}\left(\mathbf{w}\left(\mathbf{H}\left(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x})\right)\right)(\mathbf{p})\right) - \mathcal{I}^*(\mathbf{p}) \right]^2 \tag{3}$$

Due to the virtues of the parametrization it is possible to only evaluate a Jacobian, which depends on the reference image and the current image, and still achieve second order convergence [4].

3 Proposed System

An overview of the proposed system as a finite state machine (FSM) is given in Figure 5. The system starts with an initialization phase, which will be described in section 3.2. It then uses the template-based tracking algorithm to track the object as explained in section 3.3. In the event that template-based tracking fails the feature-based tracking, as described in section 3.4, is used. If the feature-based tracker is unable to recover the pose within a certain number of attempts the initialization is invoked again. Section 3.5 describes the transitions of the FSM and the reasoning behind them.

3.1 Required Information

In our system we use a textured 3D model of the object (see figure 1). This model can either be created manually or semi-automatically with commercially available products. One point to note is that it is advisable to use the same camera for texturing the model and for tracking, because this minimizes difficulties due to different image quality and image formation conditions. For the initialization registered images of the object, called keyframes, are required. They can be created directly from the textured model by rendering it from different views.

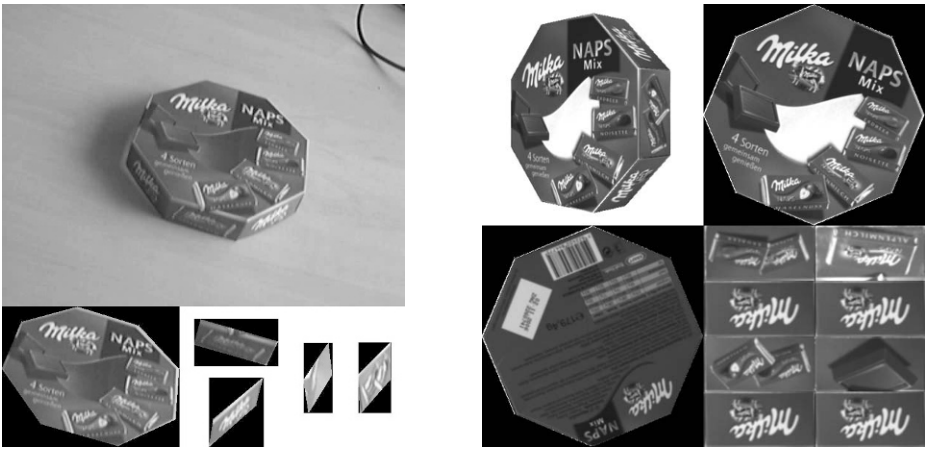


Fig. 2. Comparison of templates extracted from the video (left) and from the textured model (right). Note the low quality, the bad viewing angle and the presence of background texture in the templates extracted from the video.

3.2 Initialization

Initialization is performed using Randomized Trees. The Randomized Trees algorithm requires a reference image of the object in order to learn the appearance of the feature points. When initializing, features are extracted from the current image and matched to the features extracted in the keyframe. The pose can then be estimated from the 3D object points and corresponding 2D feature points in the current image.

Since the tracker is using a textured model of the object the accuracy of the initial pose estimation is not very critical. If on the other hand the reference templates used for tracking were extracted from the current image, the precision of the initialization procedure would be a major issue, because the quality of the tracking result depends directly on the quality of the templates used for tracking. Hence we decided to directly use the templates taken from the textured model in our system.

3.3 Template-Based Tracking

We use the ESM algorithm for template-based tracking. The object is tracked using this method until a loss of track is detected, in which case the feature-based tracker is invoked.

Reference Patch Extraction. In order to perform the tracking, the textures of the patches to be tracked are required. These textures will be called reference patches. One possibility to obtain them is to extract them from the current image. In order to do this, the pose of the object as seen in the image is required. This information is given either through the initialization or through the tracking itself, in the case that patches which had previously been occluded by the object become visible. This is however problematic since the initialization and the tracking accuracy are not always high enough to properly

extract the patch. Therefore the extracted patches often contain a some background texture. Even if the pose is accurate it is possible that the patch is partially occluded, so that the reference template will also contain this occlusion. This will eventually lead to a loss of track. Another problem is that new reference patches are often extracted under oblique viewing angles, which means that the reference patch textures will be not as informative as if they would have been seen parallel to the image plane. For these reasons we abandoned this approach and chose to use a textured model of the object to be tracked. Figure 2 shows a comparison of templates extracted from the video stream and from the textured model.

For each patch the object is rendered so that the patch is oriented parallel to the image plane. It is also important to ensure that the relative sizes of the object patches are reflected in the size of the rendered patches, since the number of pixels in a patch is directly proportional to its importance during tracking.

Since the pose parameters used to render the patches are known, the reference patches can be directly extracted from the rendered image. After this for every patch k the following information is available: The reference patch $\mathcal{I}_k^*$, the pose $\widetilde{\mathbf{T}}_k$ under which it was extracted, the patch normal $\mathbf{n}_k$ and its distance to the camera d_k. These reference patches are then reduced a few times in size by a factor of two to create a stack of reference patches at different scales, which are used to speed up the tracking in a coarse-to-fine approach.

Visibility Test. Attempting to track patches which are not visible will lead to erroneous results. Hence it is necessary to ascertain the visibility of every patch. This test is performed by rendering the model with OpenGL and using the occlusion_query extension to test which patches are visible and which are occluded. The visibility test is performed for each frame using the pose estimated in the previous frame. Thanks to the occlusion_query extension the visibility test can be performed very fast, so that it does not interfere with the tracking performance.

The Extended ESM Algorithm. We extended the formulation of the ESM algorithm as given in section 2 (see figure 3). This extension is required since in the original formulation it is implicitly assumed that all reference patches come from the same image, i.e. they were extracted in the same coordinate system. However, this is not possible when using the rendered patches, since each patch is seen under a different pose. For instance the front and back face of a cube can not be seen at the same time. Hence it would be impossible to track all the patches in the same coordinate system. This would mean that each patch had to be tracked independently without considering the constraints imposed by the object geometry. To overcome this problem the pose $\widetilde{\mathbf{T}}_k$ under which the reference patch was extracted has to be incorporated into the algorithm. This leads to the modified cost function:

$$f(\mathbf{x}) = \sum_k \sum_{\mathbf{p}_k} \left[\mathcal{I}\left(\mathbf{w}\left(\mathbf{H}\left(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x})\widetilde{\mathbf{T}}_k^{-1} \right) \right)(\mathbf{p}_k) \right) - \mathcal{I}^*(\mathbf{p}_k) \right]^2 \tag{4}$$

In order to speed up the optimization, we start at the highest scale level (lowest resolution) and optimize the cost function on this level until convergence is achieved or until the maximum number of iterations has been exceeded. If the optimization converges

The extended ESM algorithm

Input: $\mathcal{I}$, $\mathcal{I}_k^*$, $\widehat{\mathbf{T}}$, $\widetilde{\mathbf{T}}_k$
Output: $\widehat{\mathbf{T}}$

Jacobians:
$\mathbf{J}_{\mathcal{I}^*}^k = \nabla_{\mathbf{p}} \mathcal{I}_k^*(\mathbf{p})|_{\mathbf{p}=\mathbf{p}_k}$
$\mathbf{J}_{\mathcal{I}}^k = \nabla_{\mathbf{p}} \mathcal{I}\left(\mathbf{w}(\mathbf{H})\right)(\mathbf{p})|_{\mathbf{p}=\mathbf{p}_k}$
$\mathbf{J}_{\mathbf{w}} = \nabla_{\mathbf{H}}(\mathbf{w}(\mathbf{H}))(\mathbf{p})|_{\mathbf{H}=\mathbf{I}}$
$\mathbf{J}_{\mathbf{K}} = \nabla_{\mathbf{H}}(\mathbf{K}\mathbf{H}\mathbf{K}^{-1})|_{\mathbf{H}=\mathbf{I}}$
$\mathbf{J}_{\mathbf{T}}^k = \nabla_{\mathbf{T}}\mathbf{H}(\widehat{\mathbf{T}}\widetilde{\mathbf{T}}_k^{-1})^{-1}\mathbf{H}(\widehat{\mathbf{T}}\mathbf{T}\widetilde{\mathbf{T}}_k^{-1})|_{\mathbf{T}=\mathbf{I}}$
$\mathbf{J}_{\mathbf{x}} = \nabla_{\mathbf{x}}\mathbf{T}(\mathbf{x})|_{\mathbf{x}=\mathbf{0}}$

iter = 0
while (iter < max_iter)
  for each patch k
    Compute $\mathcal{I}_k = \mathcal{I}(\mathbf{w}(\mathbf{H}(\widehat{\mathbf{T}}\widetilde{\mathbf{T}}_k^{-1})))$
    for every pixel $\mathbf{p}_k$
      Compute $\mathbf{J}_k = (\mathbf{J}_{\mathcal{I}}^k + \mathbf{J}_{\mathcal{I}^*}^k)\mathbf{J}_{\mathbf{w}}\mathbf{J}_{\mathbf{K}}\mathbf{J}_{\mathbf{T}}^k\mathbf{J}_{\mathbf{x}}$
      Compute $\mathbf{y}_k = \mathcal{I}_k - \mathcal{I}_k^*$
      Append $\mathbf{J}_k$ to $\mathbf{J}$ and $\mathbf{y}_k$ to $\mathbf{y}$
  $\mathbf{x} = -2\mathbf{J}^{+}\mathbf{y}$
  if ($\|\mathbf{x}\| < \epsilon$) then exit
  $\mathbf{T}(\mathbf{x}) = \exp(\sum_{i=1}^{6} \mathbf{x}_i \mathbf{A}_i)$
  $\widehat{\mathbf{T}} = \widehat{\mathbf{T}}\mathbf{T}(\mathbf{x})$
  iter = iter+1

Fig. 3. The extended ESM algorithm

before the maximum number of iterations has been reached it is restarted on the next scale level with the pose estimated on the previous level. This is continued until the lowest scale level (highest resolution) is reached or the maximum number of iterations is exceeded.

Loss of Track. Determining when the tracker lost the object is important in order to switch to the feature-based tracking algorithm. In our system this is accomplished by computing the normalized cross correlation (NCC) between the reference patch $\mathcal{I}_k^*$ and the current patch $\mathcal{I}_k$ after the end of the optimization for all visible patches. The NCC between two patches is defined as:

$$\mathrm{NCC}(\mathcal{I}_k^*, \mathcal{I}_k) = \frac{\sum_{\mathbf{p}_k}(\mathcal{I}_k^*(\mathbf{p}_k) - \mu_k^*)(\mathcal{I}_k(\mathbf{p}_k) - \mu_k)}{N_k^2 \sigma_k^* \sigma_k} \tag{5}$$

where N_k is the number of pixels of each patch, μ_k^* and μ_k are the mean pixel intensities and σ_k^* and σ_k their standard deviations.

If the NCC of a patch falls below a certain threshold, it is excluded from the tracking. If all the patches fall below the threshold the feature-based tracker is invoked.

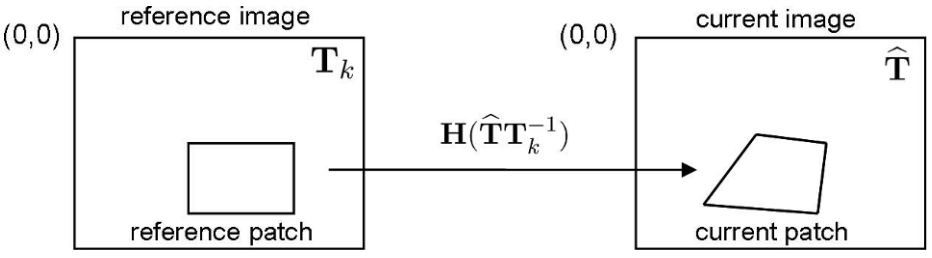


Fig. 4. Transfomations between reference patch and current patch

3.4 Feature-Based Tracking

In the event that the template-based tracker fails, the feature-based tracker is invoked. For our feature-based tracking approach we extract Harris corner points on the same reference patches used for the template-based tracking and subsequently match them to the current patch (i.e. the patch as seen in the current image) using NCC. Because NCC is not scale and rotation invariant a method had to be devised to ensure that the two patches will be seen under almost identical poses.

This is achieved as follows: Since the pose $\widetilde{\mathbf{T}}_k$ under which the reference patch k and hence the feature points were extracted is known, it is possible to determine the homography by which the current image has to be warped to obtain the reference patch. However since the object pose in the current image is not known, the pose $\widehat{\mathbf{T}}$ recovered in the previous frame is used as an approximation (see figure 4). Hence the current image has to be warped with the homography $\left(\mathbf{H}(\widehat{\mathbf{T}}\widetilde{\mathbf{T}}_k^{-1})\right)^{-1}$. Since the warping uses the pose from the previous frame the warped patch will not look exactly like the reference patch, but supposing reasonable constraints on the maximum speed of the object, it is safe to assume that the deformations will only be minor so that the NCC can still be used as a similarity measure. The feature points are then extracted in the warped patch in a window around the previous position of the patch. This reduces the computation time compared to extracting features in the whole image.

Let the matched points in the reference image and the current image be $\mathbf{p}_{k,i}$ and $\mathbf{p}'_{k,i}$ respectively. First outliers are removed using RANSAC [6]. Then the pose is estimated by minimizing the cost function:

$$f(\mathbf{x}) = \sum_k \sum_i \|\mathbf{w}\left(\mathbf{H}\left(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x})\widetilde{\mathbf{T}}_k^{-1}\right)\right)(\mathbf{p}_{k,i}) - \mathbf{p}'_{k,i}\|^2 \tag{6}$$

The parametrization is identical to that used in the template-based algorithm. Since RANSAC was already applied to remove the outliers there is no need to use a robust cost function, so a simple least-squares approach suffices.

Using the warped patches for the matching is advantageous for several reasons. First it allows the use of NCC for matching instead of a more expensive affine-invariant matching algorithm. Secondly it reduces the computational time for feature extraction, because it is only necessary to extract Harris points on the warped patch and not on the whole image. A further advantage is that this approach removes matching ambiguities in

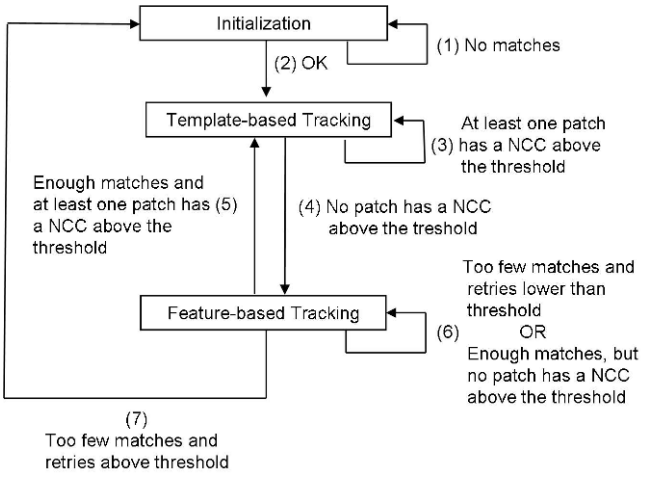


Fig. 5. Overview of the proposed tracking system

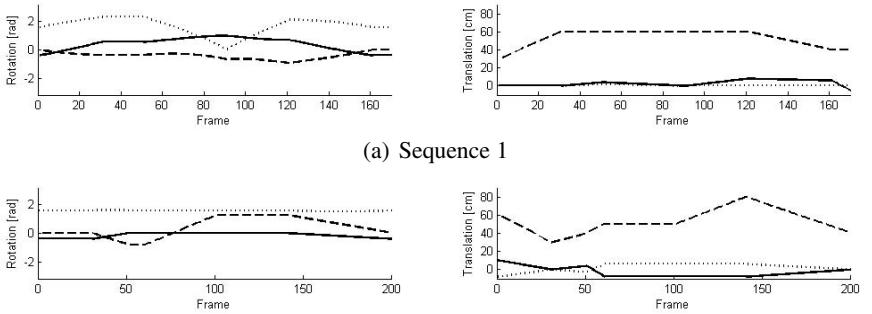
the case that multiple patches have the same texture, since by considering the previous pose only the correct patch will be used for the matching.

3.5 Finite State Machine

To decide which algorithm to use for a given frame we designed a finite state machine (see figure 5). The system starts out in the initialization phase and stays in this phase until the the object is found in the image (transition (1)). Once the object has been found we switch to the template-based tracking phase (transition (2)). The reason for starting with template-based tracking rather than with feature-based tracking is the higher accuracy and the higher frame rate, since it is possible to use a coarse-to-fine optimization approach. As long as there is at least one patch left that has a NCC higher than the threshold the template-based tracker will be used (transition (3)).

When the NCC score of all patches falls below a certain threshold the system switches to the feature-based tracker (transition (4)), because otherwise the tracking would diverge. An important issue is choosing a good threshold for the NCC. We found that a value between 0.5 and 0.7 gives the best results. For lower values the system looses track, while for higher values the feature-based approach is used most of the time, even though the template-based tracker would be faster.

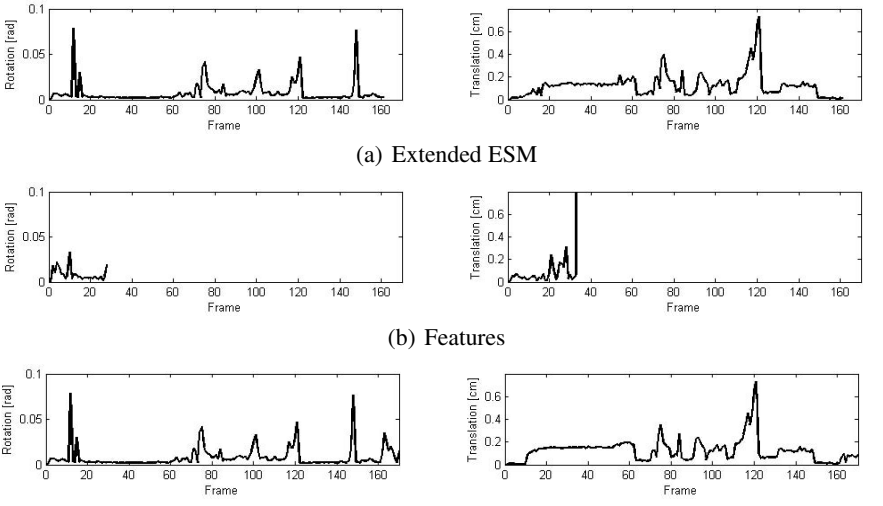
Even in the feature-based tracking phase the NCC between the reference patches and the current patches is computed. If there are enough feature matches to determine the pose, the system goes back to template-based tracking (transition (5)) unless there are no patches with a NCC above the threshold. In this case the system continues to use features (transition (6)) until at least one patch has a NCC above the threshold. If the pose cannot be recovered in the current frame the feature-based tracker is given another chance on the next few frames (transition (6)). The reason for this is that the object might just have been blurred in the current frame because of too fast motion, which makes both template-based tracking and feature extraction difficult. Often, however,



(a) Sequence 1

(b) Sequence 2

Fig. 6. Ground-truth motion in synthetic sequences (solid = x-axis, dotted = y-axis, dashed = z-axis)



(a) Extended ESM

(b) Features

(c) Proposed Approach

Fig. 7. Absolute error on synthetic sequence 1

the object slows down after a few frames, so that the feature-based tracker can find it again. If the object still cannot be found after a certain number of frames have been seen the initialization is invoked again (transition (7)).

4 Experiments

To evaluate the validity of our approach we performed several experiments on synthetic data with ground-truth and real data.

The frame rate of our system is in the range between 25 fps and 40 fps on a 1.66 GHz Intel Core-Duo CPU with 1 GB of memory. The exact value depends on a multitude

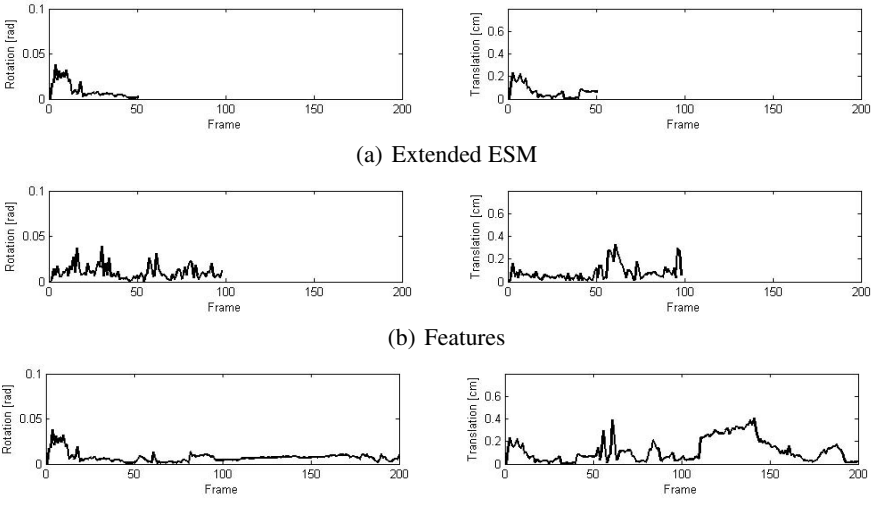


Fig. 8. Absolute error on synthetic sequence 2

of factors including the size of the reference patches, the number of scale levels, the number of feature points and the desired accuracy.

The synthetic experiments consisted of creating an animation with a textured 3D model and comparing the recovered pose parameters to the actual ones.

Figure 6 shows the ground-truth motion of one sequence with 170 and one sequence with 200 frames. There are big rotations, fast object movement and big scale changes present in both sequences. The range of the rotations is 120 degrees and the range of the translations is around 40 cm. Figure 7 and figure 8 show the absolute translation and rotation errors for the first sequence and second sequence respectively. All methods have a very small error of normally less than 3 degrees for the rotations and 4 mm for the translations. In the first sequence the extended ESM algorithm looses track at frame 162 (see figure 7(a)) due to fast object translation along the x-axis (see figure 6(a)). The feature-based algorithm already looses track much earlier at frame 31 (see figure 7(b)), because it cannot find any feature matches when the object is seen at an oblique angle. In the second sequence the feature-based algorithm performs better than the extended ESM algorithm (see figure 8). However neither algorithm can track the whole sequence. Our tracking approach on the other hand successfully tracks both sequences entirely, because it changes the tracking algorithm used at the right moment. We obtained similar results on all synthetic sequences we simulated. Since there are no blurring, illumination changes or noise in the synthetic sequences it is not possible to show how our system deals with these conditions. Therefore we also performed many real-world experiments using different objects. Figure 9 shows some experiments on real sequences made with a tea box and a candy box under varying tracking conditions. The images show how our system deals with partial occlusions, illumination changes, changes in scale and severely oblique viewing angles. This shows that the proposed algorithm is able to deal with dynamic scenarios and solve the major limitations of classical tracking algorithms

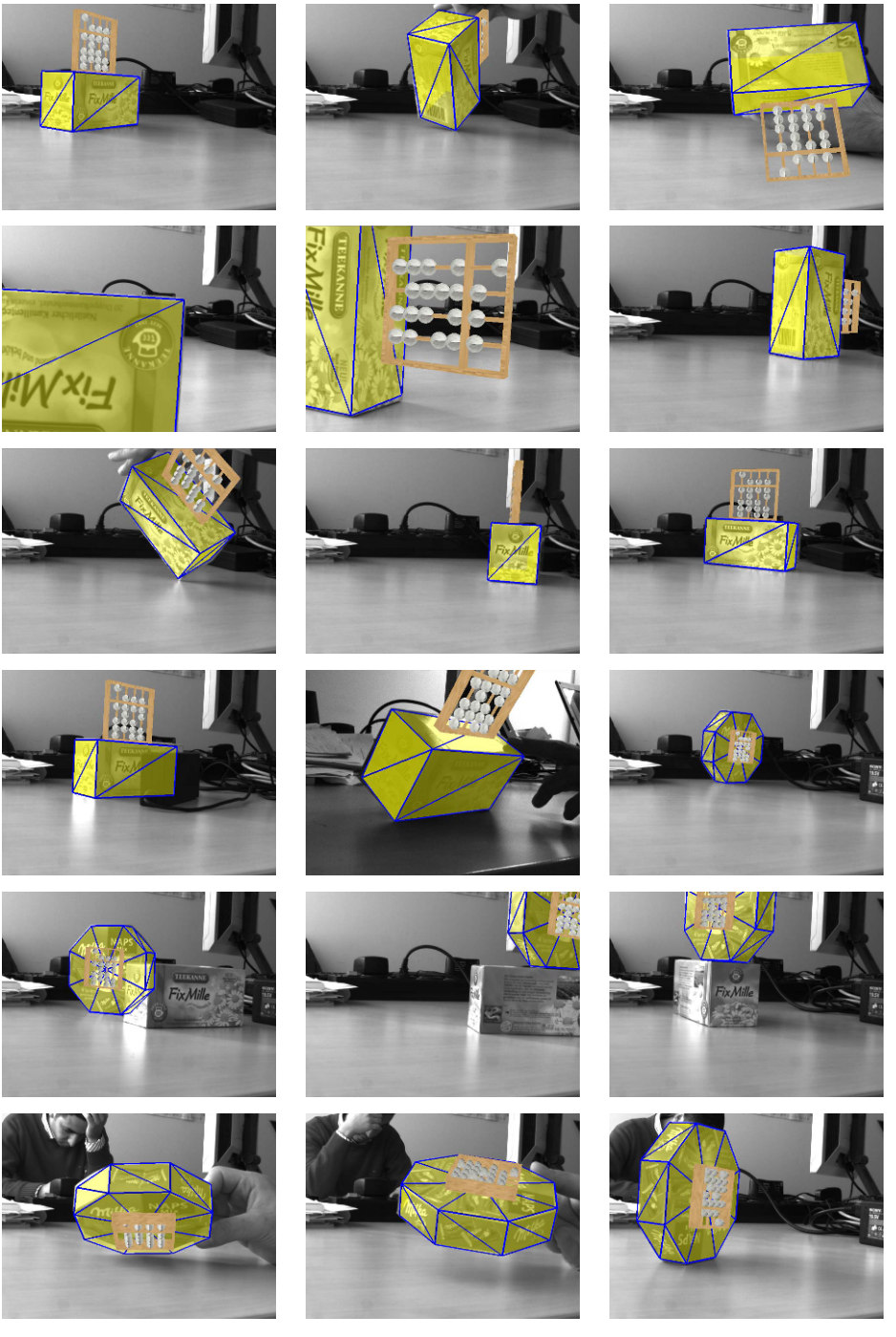


Fig. 9. Results on real data under different tracking conditions

such as partial occlusions, illumination changes and fast object movement. We can also see that it is possible to robustly overlay virtual objects in order to perform Augmented Reality.

5 Conclusions

We presented a tracking system which intelligently combines template-based and feature-based tracking. The contributions are the extension of the ESM algorithm, the formulation of the feature-based tracking and the FSM for deciding which algorithm to use for the current frame. The system has been tested on real-world sequences as well as on simulations and performs at high frame rates on a standard PC.

Compared to other algorithms proposed in the literature we achieve a higher frame rate and more robustness to fast object motions. Our approach also gives good results in the face of partial occlusions and illumination changes.

References

1. Baker, S., Dellaert, F., Matthews, I.: Aligning images incrementally backwards. Technical report, Robotics Institute, Carnegie Mellon University (2001)
2. Baker, S., Matthews, I.: Equivalence and efficiency of image alignment algorithms. In: IEEE Int. Conf. on Computer Vision and Pattern Recognition, pp. 1090–1097 (2001)
3. Bay, H., Tuytelaars, T., van Gool, L.: SURF: Speeded up robust features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3954, pp. 404–417. Springer, Heidelberg (2006)
4. Benhimane, S., Malis, E.: Real-time image-based tracking of planes using efficient second-order minimization. In: IEEE/RSJ Int. Conf. on Intelligent Robots Systems, pp. 943–948 (2004)
5. Bouthemy, P.: A maximum likelihood framework for determining moving edges. IEEE Trans. on Pattern Analysis and Machine Intelligence 11(5), 499–511 (1989)
6. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24(6), 381–395 (1981)
7. Hager, G.D., Belhumeur, P.N.: Efficient region tracking with parametric models of geometry and illumination. IEEE Trans. on Pattern Analysis and Machine Intelligence 20(10), 1025–1039 (1998)
8. Harris, C., Stephens, M.: A combined corner and edge detector. In: Proceedings of the 4th Alvey Vision Conf., pp. 147–151 (1988)
9. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, Cambridge (2004)
10. Lepetit, V., Lagger, P., Fua, P.: Randomized trees for real-time keypoint recognition. In: IEEE Int. Conf. on Computer Vision and Pattern Recognition, pp. 775–781 (2005)
11. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
12. Lucas, B., Kanade, T.: An iterative image registration technique with application to stereo vision. In: Int. Joint Conf. on Artificial Intelligence, pp. 674–679 (1981)
13. Masson, L., Dhome, M., Jurie, F.: Robust real time tracking of 3d objects. In: Int. Conf. on Pattern Recognition, pp. 252–255 (2004)

14. Pressigout, M., Marchand, E.: Real-time planar structure tracking for visual servoing: a contour and texture approach. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (2005)
15. Vacchetti, L., Lepetit, V., Fua, P.: Combining edge and texture information for real-time accurate 3d camera tracking. In: Proceedings of the Third IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 48–57 (2004)
16. Wuest, H., Vial, F., Stricker, D.: Adaptive line tracking with multiple hypotheses for augmented reality. In: Proceedings of the Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 62–69 (2005)
17. 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 (1994)

Local Structure to Solve the Correspondence Search Problem in a Monocular Pose Estimation Scenario

Marco A. Chavarria and Gerald Sommer

Cognitive Systems Group
Christian-Albrechts-University of Kiel, D-24098 Kiel, Germany
{mc,gs}@ks.informatik.uni-kiel.de
http://www.ks.informatik.uni-kiel.de

Abstract. In this paper we present a new approach that uses local structural information to find correspondences between image and model contour information. For a monocular pose estimation scenario, the pose is computed by our purposed new variant of the ICP (iterative closest point) algorithm which combines Euclidean distance with structure. A local representation of 3D free-form contours is used to get the structural information in 3D space and in the image plane. Furthermore, the local structure of free-form contours is combined with local orientation and phase obtained from the monogenic signal. With this combination, we achieve a more robust correspondence search. Our approach was tested on synthetical and real data to compare the convergence and performance of our approach against the classical ICP approach.

Keywords: Pose estimation, ICP algorithm, monogenic signal.

1 Introduction

Many actual applications in robotics and computer vision deal with objects modeled by e.g. 3D free-form contours and surfaces. Such models are widely used for problems like monocular and binocular pose estimation and object recognition among others. The more information available about the nature of these entities, the better are the chances to solve the correspondence problem in a more efficient and robust way. With respect to contour models, the simplest and most common representation in the literature uses parametric functions [17]. Active contour models, also known as "snakes" are also widely used for motion tracking and stereo matching [8].

Recently, geometric algebra [16] has been introduced in computer vision as a problem adaptive algebraic language in case of modeling geometric related problems. It turned out that the conformal geometric algebra (CGA) is especially useful because its ability of handling stratified geometrical spaces [12]. The basic geometrical entities (e.g. points, lines and planes) can be embedded in the conformal space, see [12]. Also the rigid body motion has a linear representation (called motor) with respect to all geometric entities derived from spheres. In the work of Rosenhahn [11], sets of coupled twists are used to model free-form contours and surfaces in the framework of conformal geometric algebras. In a further work, [13] the pose estimation constraints (point-line,

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 205–217, 2008.

point-plane and line-plane) were also used in that algebra. We propose a new local representation of free-form contours which allows to extract local structural information, which can be also embedded in CGA. Thus, it is also compatible with the pose estimation constraints.

Finding correspondences is one of the most challenging problems for computer vision applications. Two points correspond to each other if a similarity criteria is fulfilled. The most common and simple approach is the ICP algorithm [3]. Zhang [17] uses a modified ICP algorithm to deal with the occlusion problem. ICP algorithms combined with different metrics are also used, for example point-point [2] and point-line [5]. Chen and Medioni [4] use the sum of square distance between scene and model point in their ICP variant. An extension of this work was made by Dorai and Jain [5], where an optimal uniform weighting of points is used. A comparison of variants of the ICP algorithm is presented in [14], where the different variants are applied to align artificially generated 3D meshes. The above cited methods assume that the scene is almost aligned with the model (tracking assumption). Since these variants use as feature only point information, it is possible to optimize the algorithms for real time applications. Other methods combine the ICP algorithm with other image processing approaches like optical flow [10] or bounded Hough transform [15]. These methods seem to be robust but they are very time consuming, not suitable for real time applications. All the methods based on punctual information have to consider the tracking assumption in order to perform efficiently. For the case of the ICP variants combined with complex image processing approaches, the tracking assumption can be slightly overcome in some cases.

The basic variant of the ICP algorithm finds corresponding point pairs (image-model) by measuring the minimal Euclidean distance. In this case point coordinates can be considered as a local feature. One important question when analyzing local features is how ”local” actually the features should be. The minimal entity which can be described is a point. The only feature available is its position in the 3D space. A single point does not give much information about the object in general. From two neighbor points the local orientation can be derived and three neighbor points are enough to get local curvature. As the neighborhood is increased, more feature information can be extracted and therefore, more information about the nature of the object.

In this paper we present a new variant of structural ICP algorithm, which integrates local features (from model and image) and the structural phase information delivered from the monogenic signal [6]. One advantage of our ICP variant is that it can be perfectly applied for free-form contours and surfaces and it is robust against the tracking assumption. A local 3D contour representation is used to extract a feature set for contour segments, like concavity, convexity and straightness. In the case of the free-form surfaces, the problem is simplified by extracting the 3D silhouette with respect of the image plane during the iterative process. Once that the silhouette is projected onto the image plane, it can be considered as a planar contour and therefore the local representation can be used. With the combination of these approaches, our ICP variant reaches a compromise between computational cost and robustness against the tracking assumption.

For image feature extraction we use the monogenic scale-space approach presented by Felsberg and Sommer [7], which is briefly described in section 2. In section 3, we

introduce the local representation of 3D contours based on local motors. The feature set is obtained from a single motor and the extended set is obtained from contour segments. The ICP structural algorithm is introduced in section 4 as well as the silhouette based pose estimation algorithm for free-form surfaces. Finally, in section 5 experiments made on synthetical and real data are presented to validate the efficiency and robustness of our algorithm.

2 Image Features in Scale-Space

The monogenic scale-space representation and phase-based image processing techniques were introduced in [7]. If $p(\mathbf{x}; s)$ and $q(\mathbf{x}; s)$ are the filter responses of an image convolved with the Poisson and conjugate Poisson kernels respectively, local amplitude $a(\mathbf{x}; s)$ and phase $\mathbf{r}(\mathbf{x}; s)$ are obtained for a scale s as shown in equation (1).

$$a(\mathbf{x}; s) = \sqrt{|q(\mathbf{x}; s)|^2 + |p(\mathbf{x}; s)|^2} \qquad \mathbf{r}(\mathbf{x}; s) = \frac{q(\mathbf{X};s)}{|q(\mathbf{X};s)|}\arctan\left(\frac{|q(\mathbf{X};s)|}{p(\mathbf{X};s)}\right). \tag{1}$$

The local amplitude is related to the local energy of the signal (used to detect the presence of structure). Orientation and phase information are combined in the local phase vector. The local phase gives information about the local symmetry of the signal and the local orientation gives the orientation of the highest signal variance. An example of the monogenic response can be seen in figure 1. Then, for an edge point we chose the local features orientation and phase angles in x and y directions $F_i^{im} = \{\phi_i, \| r_i^x \|, \| r_i^y \|\}$.

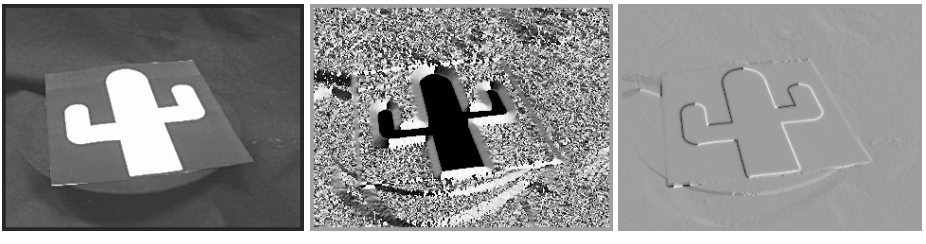


Fig. 1. Example of the monogenic signal response for an image. Original image (left), local orientation (center) and phase (right).

Once that the local amplitude and phase are obtained for a scale factor s, a contour search algorithm based on the local amplitude and orientation is applied to extract the contour segments. By changing the scale factor, low contrast edges can also be detected.

3 Local Contour Representation

The idea of the local representation is to construct a motor to approximate a contour segment. A motor is parameterized by a rotation axis and angle. This is illustrated in

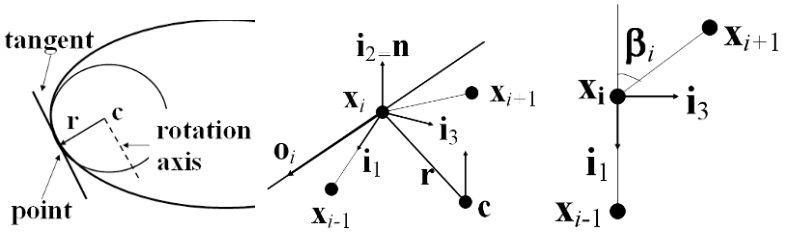


Fig. 2. Local motor for a 3D contour (left). Local coordinate system (middle) needed to get the circle parameters of the motor and the structural features (right).

the figure 2. A plane is constructed with the 3D points $\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{x}_{i+1}$, which is parameterized by its normal $\mathbf{n}$ and distance to the origin d. In that plane, a local coordinate system is defined by

$$\mathbf{i}_1 = \tfrac{\mathbf{x}_i - \mathbf{x}_{i-1}}{\|\mathbf{x}_i - \mathbf{x}_{i-1}\|},\ \mathbf{i}_2 = \mathbf{n},\ \mathbf{i}_3 = \tfrac{\mathbf{i}_1 \times \mathbf{i}_2}{\|\mathbf{i}_1 \times \mathbf{i}_2\|}. \tag{2}$$

To find the rotation axis of the motor we need to calculate the center of the circle. To make the computations easier, the problem is translated from 3D to a local coordinate system in 2D. That is the plane defined by the basis vectors $\mathbf{i}_1$ and $\mathbf{i}_3$ (see right picture of figure 2). The center of the circle $\mathbf{c}$ and the radius vector $\mathbf{r}$ are easily calculated in 2D. Then the coordinates of the center of the circle in 3D are recovered. Thus, the rotation axis of the motor in 3D is obtained with the center $\mathbf{c}$ and the normal vector $\mathbf{n}$. The rotation angle θ_i is the angle defined by the segment $\overline{\mathbf{x}_{i-1}\mathbf{c}\mathbf{x}_{i+1}}$. Finally the orientation vector $\mathbf{o}_i$ is defined by the orthogonal to the radius vector $\mathbf{r}$. Then, for every point of the 3D contour the local curvature vector and bending angle are calculated by

$$\mathbf{k}_i = (\mathbf{x}_i - \mathbf{x}_{i-1}) \times (\mathbf{x}_{i+1} - \mathbf{x}_i) \qquad \beta_i = \text{acos}\tfrac{(\mathbf{x}_i - \mathbf{x}_{i-1}) \cdot (\mathbf{x}_{i+1} - \mathbf{x}_i)}{\|(\mathbf{x}_i - \mathbf{x}_{i-1})\| \|(\mathbf{x}_{i+1} - \mathbf{x}_i)\|}, \tag{3}$$

where the points $\mathbf{x}_i$, $\mathbf{x}_{i+1}$ and $\mathbf{x}_{i-1}$ are considered in the local coordinate system. In this case, the $\mathbf{e}_3$ component of the resulting curvature vector $\mathbf{k}_i = \mathrm{x}_1\mathbf{e}_1 + \mathrm{x}_2\mathbf{e}_2 + \mathrm{x}_3\mathbf{e}_3$ changes its sign when the point is concave or convex. When the scalar x_3 has a negative sign, the point is considered locally convex. Otherwise, it will be locally concave. If the bending angle β_i has a value closed to zero, the point is considered as a part of a straight line.

An extended feature set allows to get more robust features, especially in the image plane where noise is present and digital contours are extracted. In this case we are getting features not only from a single point. The neighborhood of the point is extended to larger segments in order to take average feature values as shown in equation (4).

$$\mathbf{k}_i = \tfrac{1}{m} \textstyle\sum_{\mathrm{j}=1}^{m} \mathbf{v}_1 \times \mathbf{v}_2 \qquad \beta_i = \tfrac{1}{m} \textstyle\sum_{j=1}^{m} \text{acos}\tfrac{\mathbf{v}_1 \cdot \mathbf{v}_2}{\|\mathbf{v}_1\| \|\mathbf{v}_2\|}, \tag{4}$$

where $\mathbf{v}_1 = \mathbf{x}_i - \mathbf{x}_{i-j}$ and $\mathbf{v}_2 = \mathbf{x}_{i+j} - \mathbf{x}_i$.

By taking the point $\mathbf{x}_i$ as a reference, motors are constructed iteratively with the adjacent points. Then the contour segment is defined by the points $\{\mathbf{x}_{i-j} \cdots \mathbf{x}_{i-1}$ $\mathbf{x}_i, \mathbf{x}_{i+1} \cdots \mathbf{x}_{i+j}\ \}$ and the features of that point corresponds to the structure of the neighborhood.

3.1 3D and 2D Contour Features

We define the following structural features for a 3D point $\mathbf{x}_i$ by

$$F_i^{3D} = \{\mathbf{o}_i, \mathbf{k}_i, \beta_i\}, \tag{5}$$

where $\mathbf{o}_i$ is the local orientation vector at the point $\mathbf{x}_i$, $\mathbf{k}_i$ is the curvature vector and β_i the bending angle. To get the corresponding 2D features, the contour model points are projected onto the image plane (see figure 3), motors are constructed and the features are calculated as described in the last section with the corresponding points in image coordinates $\mathbf{x'_{i-1}}$, $\mathbf{x'_i}$ and $\mathbf{x'_{i+1}}$. The normalized orientation vector $\mathbf{o}'_i$ is obtained and its corresponding orientation angle α_i.

The concept of phase in the image plane delivers information of the local structure of the image derived from the monogenic signal. In the case of edges, the phase encodes a transition from one gray value to another in x and y directions. For 3D contour models, it is not possible to compute directly phase information in that sense. Despite of that, it is possible to assign a feature value for a projected 3D contour point that represents such transition. We call this feature transition index. Figure 3 shows the idea of transitions t_x and t_y for a point. The transition takes the values $+1$ or -1 (equivalent to the phase responses $\parallel r_i^x \parallel$ and $\parallel r_i^y \parallel$) depending on the orientation of the vector $\mathbf{o}'_i$. Thus, for a projected 3D contour point we obtain as features the orientation and transition indexes in x and y directions $F_i^{con} = \{\alpha_i, t_i^x, t_i^y\}$.

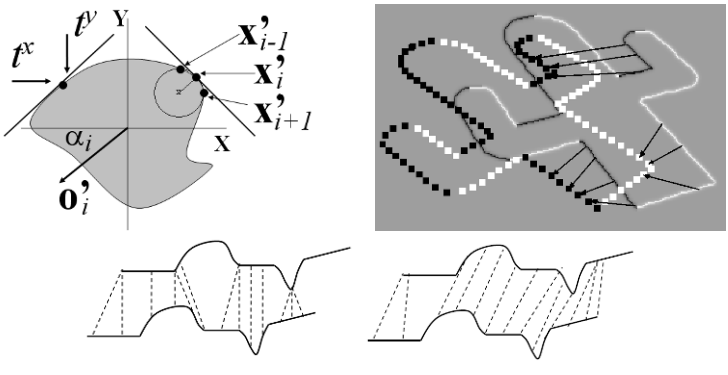


Fig. 3. Example of motor construction and the transition index in the image plane (upper left). Transition index of an projected model contour and phase response of the monogenic signal (upper right). Example of correspondence pairs for normal (bottom left) and structural (bottom right) ICP variants.

4 Structural ICP Variant

Our ICP variant combines error metrics with image feature constraints. Thus, in the image plane we have the following feature sets for projected model segments $F_i^{2Dm} = \{\alpha_i, t_i^x, t_i^y, \mathbf{k}_i^{2\mathrm{Dm}}, \beta_i^{2\mathrm{Dm}}\}$ and for detected contour segments $F_i^{2Dp} = \{\phi_i, \parallel r_i^x \parallel, \parallel$

$r_i^y \parallel, \mathbf{k}_i^{2\mathrm{Dp}}, \beta_i^{2\mathrm{Dp}}\}$. Two points (image and model) form a correspondence pair if the structural constraints are met. The phase-transition index constraint is defined as

$$C_1 = \begin{cases} 1 & \text{if } \parallel r_i^x \parallel = t_i^x \wedge \parallel r_i^y \parallel = t_i^y \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

In the following we will use the sign $\wedge$ to denote the logical "and" operation. The straightness constraint is defined from the local bending angles $\beta_i^{2\mathrm{Dm}}$ and $\beta_i^{2\mathrm{Dp}}$ as

$$C_2 = \begin{cases} 1 & \text{if } \beta_i^{2\mathrm{Dm}} < t \wedge \beta_i^{2\mathrm{Dp}} < t \\ 0 & \text{otherwise} \end{cases}, \tag{7}$$

where t is a threshold value. Finally, the concavity-convexity constraint is defined from the sign of the e_3 component of the vectors $\mathbf{k}_i^{2\mathrm{Dm}} = \mathrm{x}_1\mathbf{e}_1 + \mathrm{x}_2\mathbf{e}_2 + \mathrm{x}_3\mathbf{e}_3$ and $\mathbf{k}_i^{2\mathrm{Dp}} = \mathrm{y}_1\mathbf{e}_1 + \mathrm{y}_2\mathbf{e}_2 + \mathrm{y}_3\mathbf{e}_3$ by

$$C_3 = \begin{cases} 1 & \text{if } \mathrm{sign}(\mathrm{x}_3) = \mathrm{sign}(\mathrm{y}_3) \wedge C_2 = 0 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

Figure 3 shows the idea of ICP combined with structural constraints (straight, concave or convex). The bottom left figure shows the case where only the minimal distance is considered, on the bottom right one for the structural variant. As can be seen, for a point in the bottom curve, its corresponding point in the upper curve will be the nearest point with the same local structure. This is analogous for the ICP plus the phase-transition index constraint, see upper left picture of figure 3.

4.1 Pose Estimation for 3D Surfaces

An algorithm for pose estimation of 3D surfaces models was proposed in [10], where the 3D silhouette of the model is extracted for every iteration of the minimization process. Originally, the classical ICP algorithm was applied to find the pose of the 3D silhouette. The position of the complete surface model is updated and the process is repeated for a given number of iterations. We use a similar idea, but in our approach the problem is completely translated to the image plane by projecting the extracted 3D silhouette onto the image plane (2D silhouette). As it can be seen in figure 4, the local motors of the 2D silhouette are constructed in the image plane and its local features are also computed. The algorithm is summarized in figure 4.

5 Experiments

We used for our experiments 3D planar contour models and 3D surfaces (see figure 5) rich in structure like the "cactus" and "puzzle" models and also the "mouse" model, which has less structure. Also the power socket and motor part surface models were considered. In the first experiment we compare the convergence behavior of a normal ICP algorithm and our structural ICP variant. The initial position of the model is known, then it is translated and rotated to its actual position and projected onto the image plane to generate an artificial image. On this artificial image the corresponding contour segments and the local features are extracted. Then the pose is calculated and compared with the ground truth. For these experiments relatively large displacements were applied to the model in order to test the robustness against the tracking assumption.

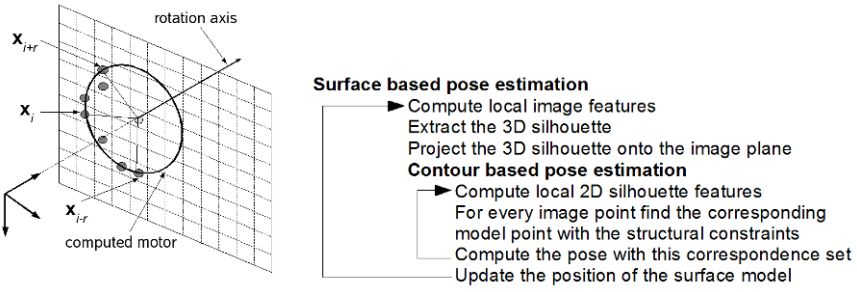


Fig. 4. Motor construction in the image plane (left). Algorithm for the silhouette based pose estimation (right).

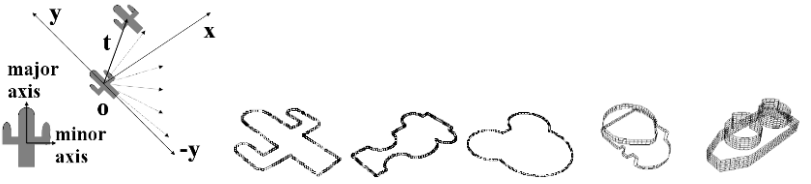


Fig. 5. The object is translated in all directions in the plane. For every translation the pose is calculated and compared with the ground truth (left). Different models used in the experiments (right to left): cactus, puzzle, mouse motor part and power socket.

5.1 Free-Form Contours

In the sequence of images in the figure 6, we compare the convergence behavior of a normal ICP algorithm against our structural variant when the tracking assumption is not met. For such cases, the pose estimation algorithm with the normal ICP variants does not converge to the actual model position. A direct comparison of the convergence behavior can be seen in the the first row of figure 7. Two different pose estimation algorithms were tested with our ICP variant, the 2D-3D [13] and projective ones [1]. In both cases, the structural ICP variants needs less iterations to converge.

The normal variants of the ICP algorithm consider as a correspondence constraint only the Euclidean distance plus a weighting error factor or a different search strategy. This has the effect that, in the first iterations many bad conditioned correspondences are found and therefore the convergence is slower or in some cases, the algorithm does not converge at all. The structural variant will also consider the constrains of equations (6), (8) and (7). This increases the probability to find better conditioned correspondences and therefore the convergence rate of the algorithm is increased.

A second experiment was made to test the robustness of our algorithm against the tracking assumption. For this case, the model was rotated around its z axis for zero to 50 degrees. As can be seen in the second row of figure 7, with the structural ICP algorithm the pose error is minimal for rotations up to 30 degrees for the 2D-3D algorithm and 40 degrees for the projective one. This shows that our structural ICP variants allows

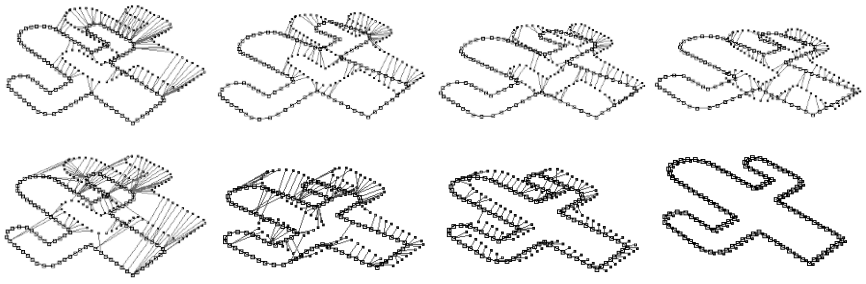


Fig. 6. Convergence sequence for normal (top row) and structural (bottom row) ICP variants applied to the cactus model

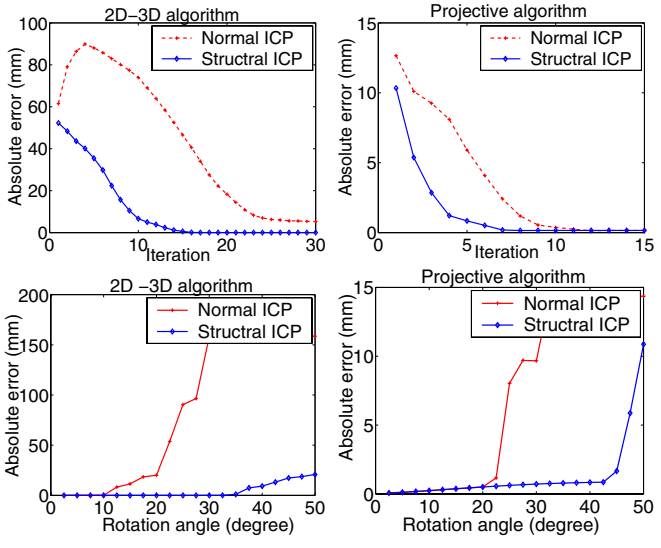


Fig. 7. First row, convergence behavior comparisons of the normal and structural ICP variants applied to the 2D-3D pose estimation algorithm (left) and the projective algorithm (right). Second row, robustness against rotations for the 2D-3D (left) and the projective algorithms (right).

larger model rotations than the normal ICP variants. The robustness of the structural ICP algorithm against the tracking assumption depends on the nature of the object and its contour. For contours which are rich in structural information larger rotations and translations are allowed.

The next experiment was made to test the magnitude and direction of the maximal possible translations allowed for the ICP structural algorithm. In this case, see figure 5, the object model was translated to all directions in the plane where it is defined. For every position, the pose was calculated with the structural ICP algorithm and the projective pose estimation [1].

The results for the cactus, puzzle and mouse models are shown in figure 8. These figures show the convergence regions of the algorithm when translations are applied. For the cactus, the algorithm is more sensitive to translations in y direction, which corresponds to translations in the major axis direction (see figure 5), while relatively large translations are allowed in x direction (minor axis direction). The same effect can be seen for the puzzle model. The figures show that for certain positions the correspondence search is better conditioned. As the translation increases, the probability to find more bad conditioned correspondences also increases and therefore the pose error. The puzzle model and the cactus are complex objects, with enough structure to deal with relatively large translations. The bottom figure shows the result for the mouse model. In this case the mouse model does not have much structural information. Therefore, as can be seen in the figure 8, for large translations the error increases considerably.

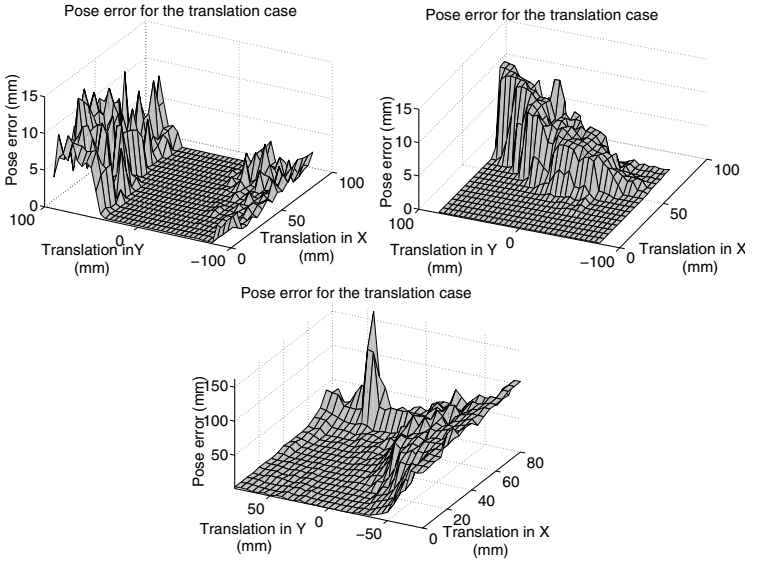


Fig. 8. Pose error for the translation case for the cactus model (top left), for the puzzle model (top right) and for the mouse model (bottom)

5.2 Free-Form Surfaces

From the initial position of the model, its main orientation axes were extracted in 3D. They define the rotation axes α, β and γ, as it can be seen in figure 9. After rotating the model around each axes, the corresponding artificial image was generated. In its new position, the pose was computed and compared with the ground truth. The upper graphics of figure 9 show a comparison of the convergence behavior for the motor and power socket models. In this case the model was rotated -30 degrees around the γ axis. The normal ICP algorithm does not converge to the ground truth pose as it can be seen

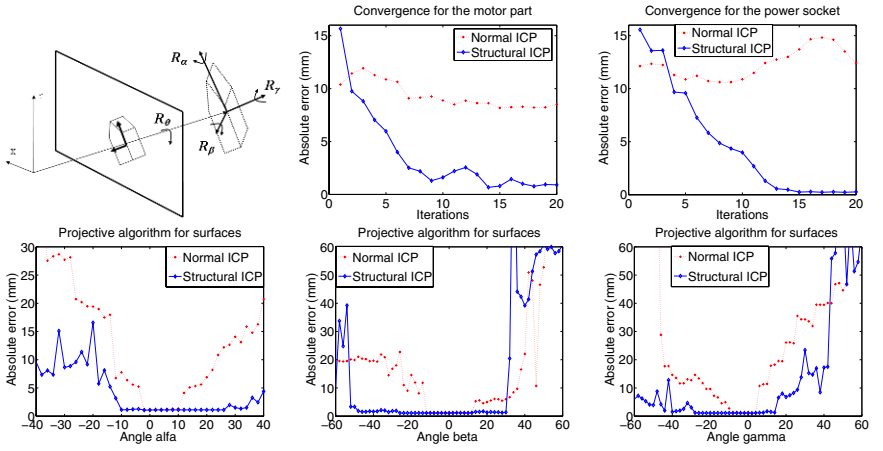


Fig. 9. Setup for the experiment for the rotation case (upper left) and convergence behavior comparison for the power socket (upper middle figure) and motor part (upper right). Convergence ranges for rotations around the axes α (bottom left), β (bottom middle figure) and γ (bottom right).

in the graphics. In contrast to that, the algorithm is able to converge to the real pose with the structural ICP.

As the model rotates around each 3D axes, its appearance changes with respect of the image plane and therefore its local structure. Because of that, it is interesting to analyze the rotation ranges within which the algorithm is still able to converge to the ground truth pose. The figure 9 also shows a comparison of the convergence ranges for the power socket model. The normal variant of the ICP algorithm converges for relatively small rotations around all axes. Whereas the structural ICP variant allows larger rotations. In the case of the rotation axes α and γ, the extracted silhouette changes drastically as the angle increases with respect of the image plane, therefore, the region where the algorithm converges is smaller. Despite of that, the the structural ICP variant shows larger convergence ranges than the classical variant for all rotation angles.

5.3 Real Pose Estimation Scenario

Finally, we applied our algorithm to image sequences of a real scenario. The algorithm was tested on a Linux based system with a 3 Ghz. Intel Pentium 4 processor. Some examples of the test sequences are shown in figure 10. The upper images show the initial position of our model and the bottom images the pose result using our ICP structural variant and the projective pose estimation algorithm. For every image the monogenic signal response was obtained and a contour search algorithm based on the local orientation and phase information was applied to detect the edge segments, then from these detected contour points the structural features were calculated. The average

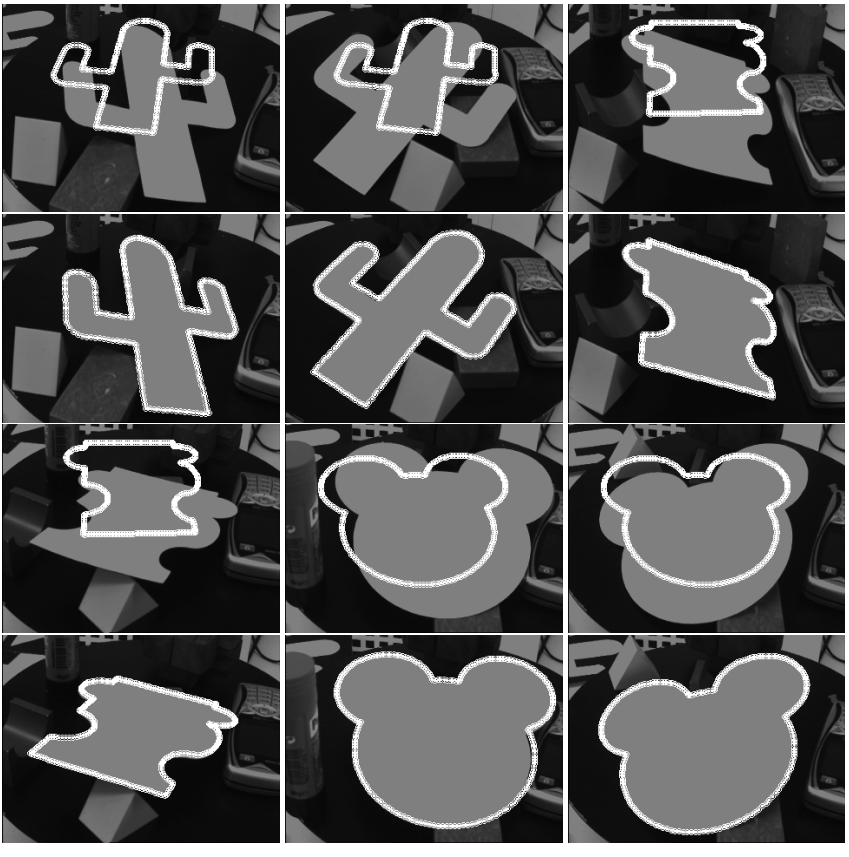


Fig. 10. Initial position (upper rows) and estimated pose (lower rows) for the cactus, puzzle and mouse models

computing time per frame for the hole image processing module was 225 milliseconds. Due to the relatively large displacement of the object, more iteration steps are needed for the algorithm to converge and therefore the computational time increases. For these sequences the average computation time (image processing plus pose estimation) was 2.65 seconds.

Figure 11 shows some examples taken from different test sequences for the 3D surfaces. Because of the extra computation of the 3D silhouette, the average computing time increases to 3.17 seconds. Presence of noise, shadows or illumination changes in the scene may cause uncertainty in the feature computation and therefore in the correspondence search. Because of that, the Euclidean distance criterion used in [9] was used to eliminate possible outliers. Correspondence pairs are rejected if their point-to-point distance is larger than 2.5 times the standard deviation of the complete correspondence set.

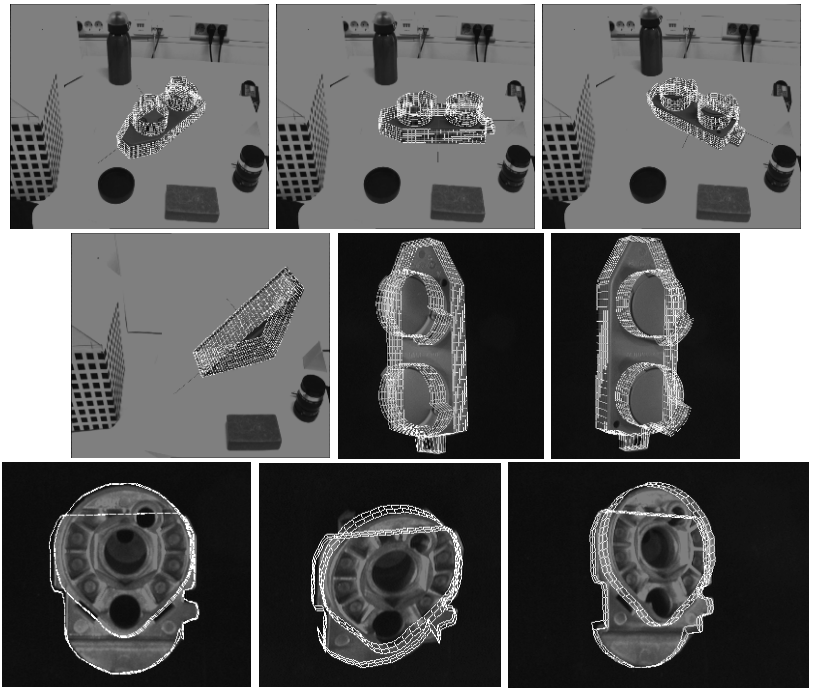


Fig. 11. Example of the estimated pose for the surface sequences with the structural ICP algorithm

6 Conclusions and Future Work

A new variant of the ICP algorithm for pose estimation of 3D free-form contours and surfaces based on local structural features was presented. The experimental test proved that our structural ICP algorithm performs efficiently for rich structured objects, for large translations and rotations between scene and object model. The experiments show that our ICP algorithm combined with the projective pose estimation approach can handle larger object displacements. That means, the feature constraints used to search correspondences and the pose estimation constraints involved in the minimization problem are better conditioned in the image plane. Although our approach does not reach requirements for real time applications [14], the computation times reported for the test sequences are a good tradeoff if we consider that the tracking assumption has been significatively overcome. A natural extension for our approach is to consider the pose estimation of free-form contours and surfaces in a more general scenarios (general occlusion and non-regular backgrounds), where local and global structural features (from model and image) will be combined to develop an approach capable to deal with even larger translations and rotation ranges.

Acknowledgments. I thank the Mexican Council of Science and Technology (CONACYT) and the German Service of Academic Exchange (DAAD) for the grant to support this project.

References

1. Araujo, H., Carceroni, R., Brown, C.: A Fully Projective Formulation to Improve the Accuracy of Lowes Pose-estimation Algorithm. Comput. Vis. Image Underst. 70(2), 227–238 (1998)
2. Benjemaa, R., Schmitt, F.: Fast Global Registration of 3d Sampled Surfaces using a Multi-z-buffer Technique. In: NRC 1997: Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling, Washington, DC, USA, pp. 113–120. IEEE Computer Society, Los Alamitos (1997)
3. Besl, P., McKay, N.: A Method for Registration of 3-d Shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence 14(2), 239–256 (1992)
4. Chen, Y., Medioni, G.: Object Modelling by Registration of Multiple Range Images. Image Vision Comput. 10(3), 145–155 (1992)
5. Dorai, C., Weng, J., Jain, A.: Optimal Registration of Object Views using Range Data. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(10), 1131–1138 (1997)
6. Felsberg, M., Sommer, G.: The Monogenic Signal. IEEE Transactions on Signal Processing 49(12), 3136–3144 (2001)
7. Felsberg, M., Sommer, G.: The Monogenic Scalespace: A unifying approach to phase-based image processing in scale-space. J. Math. Imaging Vis. 21(1), 5–26 (2004)
8. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active Contour Models. International Journal of Computer Vision 4(1), 321–331 (1987)
9. Masuda, T., Sakaue, K., Yokoya, N.: Registration and Integration of Multiple Range Rmages for 3-d Model Construction. In: ICPR 1996: Proceedings of the 1996 International Conference on Pattern Recognition (ICPR 1996), Washington, DC, USA, vol. I, p. 879. IEEE Computer Society, Los Alamitos (1996)
10. Rosenhahn, B., Brox, T., Cremers, D., Seidel, H.: A Comparison of Shape Matching Methods for Contour Based Pose Estimation. In: Reulke, R., Eckardt, U., Flach, B., Knauer, U., Polthier, K. (eds.) IWCIA 2006. LNCS, vol. 4040, pp. 263–276. Springer, Heidelberg (2006)
11. Rosenhahn, B., Perwass, C., Sommer, G.: Freeform Pose Estimation by using Twist Representations. Algorithmica 38, 91–113 (2004)
12. Rosenhahn, B., Sommer, G.: Pose Estimation in Conformal Geometric Algebra, part I: The Stratification of Mathematical Spaces. Journal of Mathematical Imaging and Vision 22, 27–48 (2005a)
13. Rosenhahn, B., Sommer, G.: Pose Estimation in Conformal Geometric Algebra, part II: Real-time Pose Estimation using Extended Feature Concepts. Journal of Mathematical Imaging and Vision 22, 49–70 (2005b)
14. Rusinkiewicz, S., Levoy, M.: Efficient Variants of the ICP algorithm. In: Proceedings of the Third Intl. Conf. on 3D Digital Imaging and Modeling, Quebec City, Canada, pp. 145–152 (2001)
15. Shang, L., Jasiobedzki, P., Greenspan, M.: Discrete Pose Space Estimation to improve ICP-based Tracking. In: 3DIM 2005: Proceedings of the Fifth International Conference on 3-D Digital Imaging and Modeling, Washington, DC, USA, pp. 523–530. IEEE Computer Society, Los Alamitos (2005)
16. Sommer, G.: Geometric Computing with Clifford Algebras. Springer, Heidelberg (2001)
17. Zhang, Z.: Iterative Point Matching for Registration of Free-form Curves and Surfaces. Int. J. Comput. Vision 13(2), 119–152 (1994)

Disparity Contours – An Efficient 2.5D Representation for Stereo Image Segmentation*

Wei Sun[1] and Stephen P. Spackman[2]

[1] Intel Corporation
2200 Mission College Blvd., Santa Clara, CA 95054, U.S.A.
wei.sun@mail.mcgill.ca
[2] Quantum Corporation
1650 Technology Drive, Suite 800, San Jose, CA 95110, U.S.A.
stephen@acm.org

Abstract. *Disparity contours* are easily computed from stereo image pairs, given a known background geometry. They facilitate the segmentation and depth calculation of multiple foreground objects even in the presence of changing lighting, complex shadows and projected video background. Not relying on stereo reconstruction or prior knowledge of foreground objects, a disparity contour based image segmentation method is fast enough for some real-time applications on commodity hardware. Experimental results demonstrate its ability to extract object contour from a complex scene and distinguish multiple objects by estimated depth even when they are partially occluded.

Keywords: Multi-object segmentation, stereo matching, background model, disparity verification, disparity contours.

1 Introduction

A wide variety of applications require an efficient method to extract moving objects from a scene. One particular case, of isolating and distinguishing multiple objects in the face of rapid changes in illumination and texture, is especially relevant to augmented reality, immersive telepresence, and the entertainment and film industry, where projected moving backgrounds are often present. An example is shown in Fig. 1, where local and remote users interact with each other and with virtual objects in a virtual world. To successfully track, render and interact with users in this synthetic environment, the system must separate them visually from their actual physical surroundings in real time.

This problem motivated the development of *disparity contours*, a simple and easily computed 2.5D representation from which object segmentation and depth computation can be derived.

Existing approaches to object/background segmentation fall into two broad categories, depending on how many views they take as input. Single view background subtraction [24,16,21,14,15,2] compares each image to a reference model and labels

* This research was carried out at the Centre for Intelligent Machines, McGill University, Montreal, Quebec, Canada.

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 218–231, 2008.

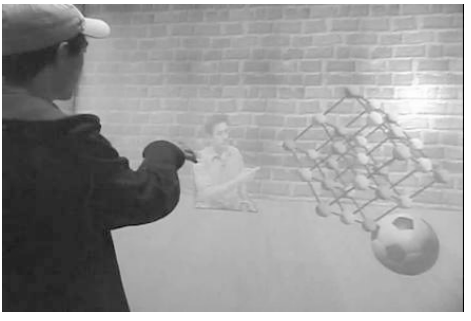


Fig. 1. Example of an augmented reality environment

pixels as background or foreground based on statistics. Despite their adaptability to slow changes in lighting, texture, geometry and shadow, methods of this type all assume background change to be much less dynamic than foreground.

Using occlusion based depth ordering, layered motion segmentation [22,8,1,23] decomposes image sequences into sets of overlapping layers, each described by a smooth optical flow field. Discontinuities in the description are attributed to moving occlusions, resulting in a (weak) 2.5D scene representation. Unfortunately, the computation of optical flow is time-consuming, and these methods cannot distinguish a real scene from a video background.

Integrating multiple views is a natural alternative for tackling dynamic environments, with added benefits in handling occlusion. Some systems work from 3D reconstruction to object segmentation and tracking [13], and others combine segmentation with stereo matching [20,12]. Sadly, frame-by-frame stereo reconstruction is also slow and so far unsuited to real-time use. Moreover, the uniform or repetitive textures common in indoor scenes and video-augmented spaces constitute worst-case inputs for stereo matching algorithms [11,17], often leading to disappointing results.

Attempts have been made to use stereo while limiting computational cost. One of them combines stereo with background subtraction and suggests disparity verification for segmentation under rapid illumination change [7]. Using three cameras on wide baselines, the method constructs offline disparity mappings for the background images, and at runtime separates foreground from background by matching pixels corresponding in the mappings, thus avoiding slow disparity search. Unfortunately, the wide baseline setup, despite its effectiveness in extracting the entire foreground area, has difficulty fusing multiple views of a target, which is essential for tracking multiple moving objects. This weakness, in the longer run, also limits the method's adaptability to background geometry change.

Another approach increases speed by decreasing the number of disparity layers in stereo matching, and proposes layered dynamic programming and layered graph cut for foreground/background separation [10]. Although tolerance of background motion has been demonstrated, published results show only cases with a substantial depth difference between background and foreground, with foreground objects very close to the camera. This is a strong limitation for many real-world applications.

Both of these fast stereo approaches stop at bi-layer pixel labelling, and do not attempt to distinguish multiple objects. The additional processing required for accurate object location would be extensive.

This paper introduces a new small-baseline stereo representation, disparity contours, computed by geometrically informed background subtraction. This representation directly provides object boundaries and allows fast, incremental disparity adjustment for objects at different depths, leading to a straightforward depth extraction method. On this basis, we have developed a stereo segmentation system that can isolate and distinguish multiple objects in the presence of highly dynamic lighting and background texture. In addition to the advantage of bypassing full stereo reconstruction and achieving fast performance, it has the potential to support 2D and 3D object tracking and background geometry update.

2 Disparity Contours

In this section, we explain disparity contours in detail, showing how to use them to estimate foreground disparity and depth, and verify object hypotheses.

2.1 Background Hypothesis Falsification (BHF)

In our proposal, the spatial geometry of a background is represented by a **background disparity map** (BDM) describing the relative displacement, or **disparity**, of pixels corresponding to the same background point in each camera view. The input images are first undistorted and rectified so that pairs of conjugate epipolar lines become colinear and horizontal [6]. This brings the pixels originating from a scene point s to a common scanline, falling at $(\mathbf{x}_{\mathrm{L}}(s), \mathbf{y}(s))$ in V_{L}, the left view, and $(\mathbf{x}_{\mathrm{R}}(s), \mathbf{y}(s))$ in V_{R}, the right. We call the difference

$$\mathbf{x}_{\mathrm{L}}(s) - \mathbf{x}_{\mathrm{R}}(s) = \mathbf{d}_{\mathrm{B}}(s), \tag{1}$$

which increases with proximity to the camera, the **background disparity** at s, and define the BDM to be

$$\mathrm{BDM} = \left\{ \langle \mathbf{x}_{\mathrm{L}}(s), \mathbf{x}_{\mathrm{R}}(s), \mathbf{y}(s) \rangle \right\}, \tag{2}$$

where s ranges over all background scene points visible to either camera and within their common field of view.

Given the BDM for two cameras, each new pair of captured images are hypothesized to be of background alone, and a **view difference map** (VDM) is computed from the stored correspondences by block matching, using a vertical stripe window to maintain contour widths and aggregate neighborhood support:

$$\mathrm{VDM}_{\mathrm{BHF}}(x_{\mathrm{L}}, x_{\mathrm{R}}, y) = \sum_{u,v} \left| \mathrm{V}_{\mathrm{L}}(x_{\mathrm{L}} + u, y + v) - \mathrm{V}_{\mathrm{R}}(x_{\mathrm{R}} + u, y + v) \right|, \quad \text{where } \langle x_{\mathrm{L}}, x_{\mathrm{R}}, y \rangle \in \mathrm{BDM} \text{ and } y_{\mathrm{L}} = y_{\mathrm{R}} = y. \tag{3}$$

If the images are well synchronized, this operation cancels instantaneous background texture.

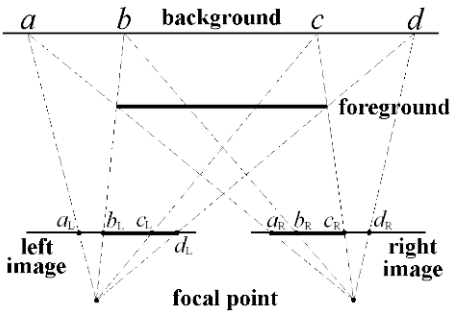


Fig. 2. Background hypothesis falsification. Mismatch occurs at object boundaries and interiors.

Thus, ideally, $\mathrm{VDM}_{\mathrm{BHF}}(x_{\mathrm{L}}, x_{\mathrm{R}}, y) = 0$ where a scene point s is truly part of the background, but is larger if either of the pixels $\mathrm{V}_{\mathrm{L}}(x_{\mathrm{L}}, y)$ and $\mathrm{V}_{\mathrm{R}}(x_{\mathrm{R}}, y)$ belongs to a foreground object. Thus a value significantly different from zero leads to the falsification of the hypothesis that the BDM is an accurate local description at a given scene point.

In reality, the result depends on the visual difference between background and foreground, between different foreground objects, and between points within foreground objects, as illustrated in Fig. 2. Suppose $\langle a_{\mathrm{L}}, a_{\mathrm{R}}, y\rangle$, $\langle b_{\mathrm{L}}, b_{\mathrm{R}}, y\rangle$, $\langle c_{\mathrm{L}}, c_{\mathrm{R}}, y\rangle$ and $\langle d_{\mathrm{L}}, d_{\mathrm{R}}, y\rangle$ are entries on the y^{th} scanline in the BDM. At object boundaries, segments $[(a_{\mathrm{L}}, y), (b_{\mathrm{L}}, y)]$ and $[(c_{\mathrm{L}}, y), (d_{\mathrm{L}}, y)]$ in the left image are mismatched against segments $[(a_{\mathrm{R}}, y), (b_{\mathrm{R}}, y)]$ and $[(c_{\mathrm{R}}, y), (d_{\mathrm{R}}, y)]$ in the right, respectively. In object interiors, segment $[(b_{\mathrm{L}}, y), (c_{\mathrm{L}}, y)]$ is mismatched against segment $[(b_{\mathrm{R}}, y), (c_{\mathrm{R}}, y)]$.

As most (non-camouflaged) real-world objects are texturally coherent, we find that foreground-background mismatches at object boundaries have higher intensity than those from object interior autodecorrelation, as visible in Fig. 9(b). Further, since boundary mismatches derive from the geometry of projection, they also have more regular shape. We now examine these boundary mismatches in detail.

2.2 Disparity Contours

At object boundaries, stereo mismatch arising from background hypothesis falsification forms contours, as illustrated in Fig. 3. Since disparity increases with proximity to the cameras, the width of the contour area in which background is mismatched against foreground depends on how poor the assumption of background was, in terms of depth error.

Consider, without loss of generality, the left view. According to Fig. 2 and Eq. (1), we have

$$\mathbf{x}_{\mathrm{L}}(a_{\mathrm{L}}) - \mathbf{x}_{\mathrm{R}}(a_{\mathrm{R}}) = \mathbf{d}_{\mathrm{B}}(a_{\mathrm{L}}). \tag{4}$$

Since b_{L} and a_{R} map to the same foreground point,

$$\mathbf{x}_{\mathrm{L}}(b_{\mathrm{L}}) - \mathbf{x}_{\mathrm{R}}(a_{\mathrm{R}}) = \mathbf{d}_{\mathrm{F}}(b_{\mathrm{L}}), \tag{5}$$

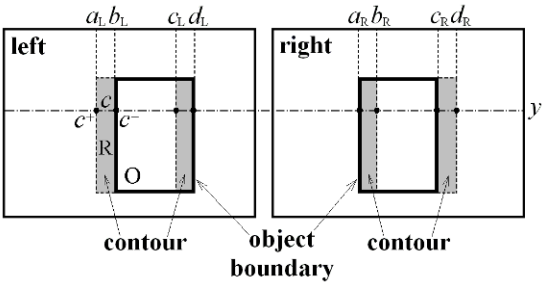


Fig. 3. Disparity contours from background hypothesis falsification. Contour widths equal differential disparities between object and background.

where $\mathbf{d}_{\mathrm{F}}(b_{\mathrm{L}})$ is the **foreground disparity** at b_{L}. Subtracting Eq. (4) from (5) and eliminating $\mathbf{x}_{\mathrm{R}}(a_{\mathrm{R}})$,

$$\mathbf{x}_{\mathrm{L}}(b_{\mathrm{L}}) - \mathbf{x}_{\mathrm{L}}(a_{\mathrm{L}}) = \mathbf{d}_{\mathrm{F}}(b_{\mathrm{L}}) - \mathbf{d}_{\mathrm{B}}(a_{\mathrm{L}}). \tag{6}$$

Similarly,

$$\mathbf{x}_{\mathrm{L}}(d_{\mathrm{L}}) - \mathbf{x}_{\mathrm{L}}(c_{\mathrm{L}}) = \mathbf{d}_{\mathrm{F}}(d_{\mathrm{L}}) - \mathbf{d}_{\mathrm{B}}(c_{\mathrm{L}}). \tag{7}$$

This means that the lengths of the segments $[(a_{\mathrm{L}}, y), (b_{\mathrm{L}}, y)]$ and $[(c_{\mathrm{L}}, y), (d_{\mathrm{L}}, y)]$ are exactly the differences between the foreground and background disparities at the object boundaries, and encode depth. Combining such segments vertically as in Fig. 3 will yield the depth-encoding contours of foreground objects, referred to as **disparity contours**. As the figure makes clear, the resulting contours lie at the left of object boundaries in the left image but at the right in the right image. There is thus no ambiguity in the boundary locations once the contours are extracted.

2.3 Foreground Disparity and Depth Estimation

Foreground disparity can be estimated given the extracted disparity contours and the background disparities. Let c be a contour line segment in the left view of length $|c|$, and c^+ and c^- its left and right end points, as in Fig. 3. From Fig. 2 and Eq. (6), we have

$$\mathbf{d}(c) = |c| = \mathbf{x}_{\mathrm{L}}(c^-) - \mathbf{x}_{\mathrm{L}}(c^+) = \mathbf{d}_{\mathrm{F}}(c^-) - \mathbf{d}_{\mathrm{B}}(c^+). \tag{8}$$

Here $\mathbf{d}(c)$ is the **differential disparity** between the background and foreground. We rewrite this equation, simplifying the notation without ambiguity, as:

$$\mathbf{d}_{\mathrm{F}}(c) = \mathbf{d}_{\mathrm{B}}(c) + \mathbf{d}(c), \tag{9}$$

which yields the foreground disparity at the boundary point. Let R be a contour region containing $|\mathrm{R}|$ such line segments. The average foreground disparity of R can be calculated by:

$$\bar{\mathbf{d}}_{\mathrm{F}}(\mathrm{R}) = |\mathrm{R}|^{-1} \sum_{c \in \mathrm{R}} \mathbf{d}_{\mathrm{F}}(c). \tag{10}$$

Similarly, the average disparity of an object O is:

$$\bar{\mathbf{d}}_{\mathrm{F}}(\mathrm{O}) = \frac{\sum_{\mathrm{R}\in\mathrm{O}} |\mathrm{R}|\bar{\mathbf{d}}_{\mathrm{F}}(\mathrm{R})}{\sum_{\mathrm{R}\in\mathrm{O}} |\mathrm{R}|}. \tag{11}$$

Thus, its average depth can be computed as

$$\bar{\mathbf{z}}_{\mathrm{F}}(\mathrm{O}) = b \cdot \alpha \;/\; \bar{\mathbf{d}}_{\mathrm{F}}(\mathrm{O}), \tag{12}$$

where b is the baseline, i.e. the distance between the focal points of the two cameras, and α is the pixel focal length along the x axis of the (virtual) rectified cameras [18,6].

2.4 Foreground Hypothesis Verification (FHV)

Once disparity contours are extracted, we need to verify the potential objects delimited by the contours. Again, this can be done using disparity verification.

Let R_i and R_j be two vertically overlapping contour regions in the left view, i.e. $\boldsymbol{\pi}_y(\mathrm{R}_i) \cap \boldsymbol{\pi}_y(\mathrm{R}_j) \neq \emptyset$, as shown in Fig. 4. If there is a potential foreground object between R_i and R_j, and assuming the depth range of a foreground object is much smaller than the object-to-camera distance, its average disparity can be approximated, based on Eq. (11), by

$$\bar{\mathbf{d}}_{\mathrm{F}}(\mathrm{R}_i, \mathrm{R}_j) = \frac{|\mathrm{R}_i|\,\bar{\mathbf{d}}_{\mathrm{F}}(\mathrm{R}_i) + |\mathrm{R}_j|\,\bar{\mathbf{d}}_{\mathrm{F}}(\mathrm{R}_j)}{|\mathrm{R}_i| + |\mathrm{R}_j|}. \tag{13}$$

This foreground hypothesis can be verified by

$$\mathrm{VDM}_{\mathrm{FHV}}(x_{\mathrm{L}}, x_{\mathrm{R}}, y) = \left|\mathrm{V}_{\mathrm{L}}(x_{\mathrm{L}}, y) - \mathrm{V}_{\mathrm{R}}(x_{\mathrm{R}}, y)\right|, \quad \text{where } (x_{\mathrm{L}}, y) \in \mathrm{A}_{ij} \text{ and } x_{\mathrm{L}} - x_{\mathrm{R}} = \bar{\mathbf{d}}_{\mathrm{F}}(\mathrm{R}_i, \mathrm{R}_j). \tag{14}$$

For robustness, a **contour matching cost** is defined to normalize this result over the matching area A_{ij}:

$$\boldsymbol{\mu}_{\mathrm{M}}(\mathrm{R}_i, \mathrm{R}_j) = \frac{\sum_{(x_{\mathrm{L}},y)\in\mathrm{A}_{ij}} \mathrm{VDM}_{\mathrm{FHV}}(x_{\mathrm{L}}, x_{\mathrm{L}} - \bar{\mathbf{d}}_{\mathrm{F}}(\mathrm{R}_i, \mathrm{R}_j), y)}{\mathrm{area}\,(\mathrm{A}_{ij})}. \tag{15}$$

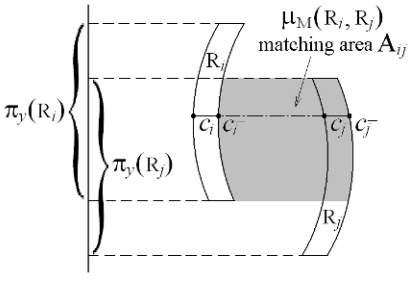


Fig. 4. Foreground hypothesis verification by matching two disparity contour regions R_i and R_j, left view

Then the foreground hypothesis between R_i and R_j is confirmed if $\boldsymbol{\mu}_{\mathrm{M}}(\mathrm{R}_i, \mathrm{R}_j)$ is less than a threshold τ_{M}. Similarly, if an object is formed by grouping several contour regions, an **object cost** $\boldsymbol{\mu}_{\mathrm{M}}(\mathrm{O})$ can be defined using the object's peripheral contours on the left and right sides to verify the object hypothesis.

2.5 Contour Grouping Direction

§2.4 provides an analysis of the ideal case of foreground verification. In reality, both background hypothesis falsification (BHF) and foreground hypothesis verification (FHV) depend on the amount of texture in the foreground and background. To understand the matter further, Fig. 5 plots the left view VDM results of BHF, FHV, and their subtraction along the y^{th} scanline of Fig. 3, assuming significant visual difference between foreground and background. The results are classified according to whether the background and foreground are textured or plain.

As can be observed, BHF distinguishes the foreground from the background only if the foreground is textured, and FHV does so only if the background is textured. However, the subtraction of the two, BHF−FHV, yields consistently higher values within the object area, between (b_{L}, y) and (c_{L}, y), than in the background area, left of (a_{L}, y) and right of (d_{L}, y), in three of the four cases. Even though the last case, where both the foreground and background are plain, would pose difficulties, it is statistically rare that the entire background and foreground areas remain textureless over time in a real environment. Therefore, by comparing BHF−FHV values in the left and right neighbourhoods of a contour region, we are able to determine in which direction, left or right, a contour region should be grouped with other contours.

Let the left neighbourhood $\mathrm{A}_{\sqsubset}(\mathrm{R})$ of a contour region R be constrained by both the rightmost vertically overlapping contour region to the left of R and a distance threshold τ_{N}, whichever is closer, as illustrated in Fig. 6.

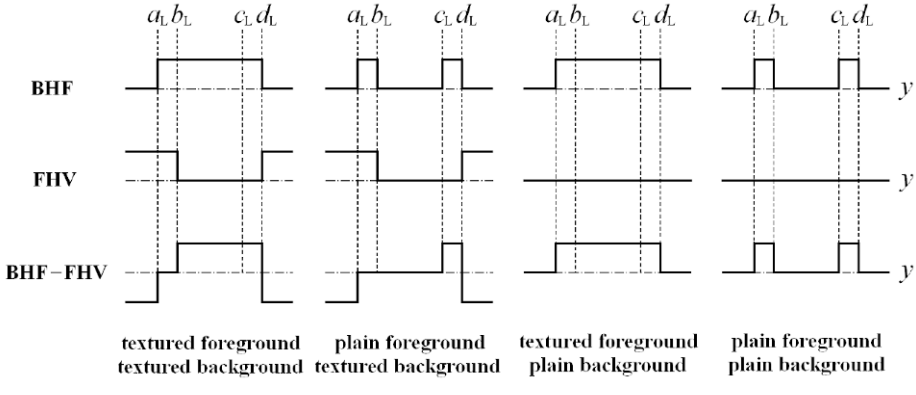


Fig. 5. The y^{th} scanline of view difference map (VDM), left view, where (a_{L}, y), (b_{L}, y), (c_{L}, y), (d_{L}, y) are the end points of disparity contour line segments, as in Fig. 3. The horizontal dashed lines indicate zero values and the solid lines indicate the VDM calculation results, simplified as positive, negative or zero. Top: $\mathrm{VDM}_{\mathrm{BHF}}$ from background hypothesis falsification; middle: $\mathrm{VDM}_{\mathrm{FHV}}$ from foreground hypothesis verification; bottom: $\mathrm{VDM}_{\mathrm{BHF}} - \mathrm{VDM}_{\mathrm{FHV}}$.

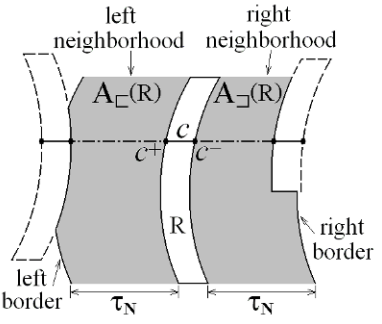


Fig. 6. Left and right neighbourhoods of contour region R for computing contour grouping direction

Let $\boldsymbol{\mu}_{\sqsubset}(\mathrm{R})$ denote the normalized subtraction result BHF−FHV in $\mathrm{A}_{\sqsubset}(\mathrm{R})$:

$$\boldsymbol{\mu}_{\sqsubset}(\mathrm{R}) = \frac{\sum_{(x_\mathrm{L},y)\in \mathrm{A}_{\sqsubset}(\mathrm{R})} \mathrm{VDM}_{\mathrm{BHF}}(x_\mathrm{L}, x'_\mathrm{R}, y) - \mathrm{VDM}_{\mathrm{FHV}}(x_\mathrm{L}, x''_\mathrm{R}, y)}{\mathrm{area}\,(\mathrm{A}_{\sqsubset}(\mathrm{R}))},$$
$$\text{where } \langle x_\mathrm{L}, x'_\mathrm{R}, y\rangle \in \mathrm{BDM} \text{ and } x_\mathrm{L} - x''_\mathrm{R} = \bar{\mathbf{d}}_\mathrm{F}(\mathrm{R}). \quad (16)$$

Similarly, let $\boldsymbol{\mu}_{\sqsupset}(\mathrm{R})$ denote the normalized BHF−FHV result in R's right neighbourhood $\mathrm{A}_{\sqsupset}(\mathrm{R})$. The **contour grouping direction** $\boldsymbol{\mu}_\mathrm{D}(\mathrm{R})$ can then be calculated as:

$$\boldsymbol{\mu}_\mathrm{D}(\mathrm{R}) = \boldsymbol{\mu}_{\sqsubset}(\mathrm{R}) - \boldsymbol{\mu}_{\sqsupset}(\mathrm{R}). \quad (17)$$

According to Fig. 5, R is on the left boundary of an object if $\boldsymbol{\mu}_\mathrm{D}(\mathrm{R}) < 0$, on the right boundary if $\boldsymbol{\mu}_\mathrm{D}(\mathrm{R}) > 0$, and within an object or background (or both the object and background are textureless) if $\boldsymbol{\mu}_\mathrm{D}(\mathrm{R}) = 0$.

3 Multi-object Segmentation

Based on disparity contours, a multi-object segmentation system has been developed, as illustrated in Fig. 7; its implementation is discussed in depth elsewhere [19].

The system factors the segmentation problem into two stages: a well-understood offline stage and a novel online one.

Using the parameters of two calibrated cameras [18], the offline stage constructs a background geometry model in the form of a background disparity map (BDM). This can be done by stereo matching [11,12,17], structured light [25], or ray tracing [5] from direct measurements of room geometry. The result is shown in Fig. 8.

Following the ideas of §2, the online stage compares new frames, captured, synchronized, undistorted and rectified, according to the pixel correspondence stored in the BDM, to falsify the background hypothesis of the scene (BHF) and generate difference images, as shown in Fig. 9(b).

To extract disparity contours in a difference image D, a simple $[-1\ 1]$ edge operator is applied to generate an edge image. Clearly, positive edges are obtained on the left of

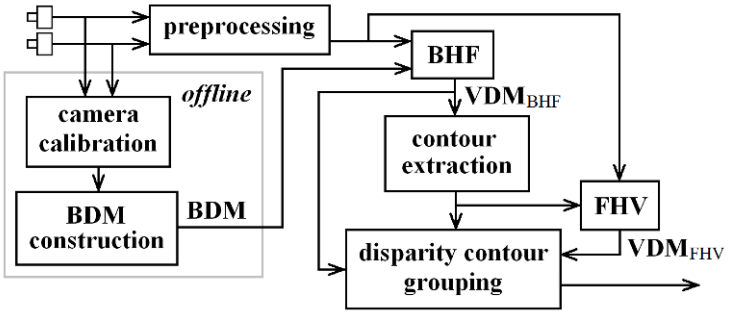
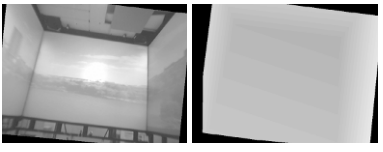
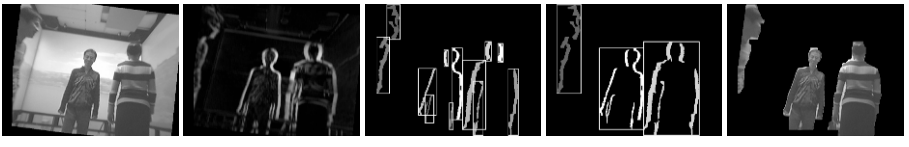


Fig. 7. Multi-object segmentation system overview



(a) (b)

Fig. 8. (a) Left background image after undistortion and rectification. (b) Background disparity map (BDM) by ray tracing from 3D measurements.



(a) (b) (c) (d) (e)

Fig. 9. Processing sequence. (a) Original left view $\mathrm{V_L}$ after undistortion and rectification. (b) Corresponding left projection $\mathrm{D_L}$ of view difference map. High responses occur at object boundaries and within objects of non-uniform texture. (c) Extracted and cleaned disparity contours, augmented with contour region bounding boxes. Brightness represents average region intensity. (d) Contours grouped by matching and disparity verification. (e) Objects segmented by grouped contours.

a contour ridge in D and negative edges on the right. Positive and negative edge points are thus paired to form horizontal line segments, c, and their lengths, $|c| = \mathbf{d}(c)$, equal the differential disparities between foreground and background. Contour regions, R, are then formed by connecting these line segments vertically.

In order to remove noise caused by background model inaccuracies and foreground object internal texture, extracted contours go through an outlier removal step, including local line segment regularization and global region outlier removal. First, based on a Gaussian assumption, the horizontal line segments within each contour region whose lengths are outliers with respect to the region average are eliminated. Contour regions

thus disconnected are reconnected by interpolation. Second, based on the observation that unwanted contour regions due to noise are usually small and of low intensity, and again assuming a Gaussian distribution for the two variables, the regions whose area and intensity are outliers with respect to the largest and brightest region are removed. Fig. 9(c) shows the final cleaned contours.

Computing closed bounding object contours from bounding fragments relies on contour grouping, studied for many decades in perceptual organization [3,4]. However, reliable contour grouping requires much computation and is unsuitable for real-time applications. We use a simple technique based on contour matching and disparity verification.

First, an initial grouping is performed to associate a contour region to its neighbours if they are close to each other and have similar average intensity and disparity. Then, based on contour grouping direction $\boldsymbol{\mu}_{\mathrm{D}}(\mathrm{R})$, neighbouring contour regions with appropriate directions are selected for matching. If the matching cost $\boldsymbol{\mu}_{\mathrm{M}}(\mathrm{R}_i, \mathrm{R}_j)$ is low, the regions are labelled to the same group. Finally, after all contour regions are grouped to objects, the object costs $\boldsymbol{\mu}_{\mathrm{M}}(\mathrm{O})$ are evaluated and objects with high cost are eliminated as false. Fig. 9(d) demonstrates the result of contour grouping.

As explained in §2.2, disparity contours contain information about object boundary location in the input images. Therefore, objects can be segmented using the grouped contours, as shown in Fig. 9(e).

4 Results and Analysis

The proposed segmentation system was tested in the augmented reality environment shown in Fig. 8(a). This space, which is representative of an important class of target environments, has a simple geometry, easing BDM construction, and allows for dynamic re-texturing of over 80% of the wall surface. Since, however, the method depends only on geometric stability, it is also applicable to more complex scenes.

A video with rapid changes in texture and illumination was projected onto the three screens surrounding the subjects. Two cameras on a small baseline were used and a grayscale image sequence of 1130 frames containing over 2000 foreground object instances was captured. Sample images are shown in Fig. 10. As can be seen, the proposed method extracts multiple foreground objects despite complex changes in background texture.

In order to study the accuracy of object location, a quantitative analysis was conducted based on object bounding boxes, as shown in Table 1 and Fig. 11. The rate of 'accurate' object location, indicated by exact bounding boxes, with respect to the number of total objects reaches 60%, while the rate of 'correct' object location, including exact, noise enlarged, and partial object bounding boxes, totals 85%. Although partial occlusion, resulting in irregular contours, poses a challenge, the system still yields nearly 40% for 'accurate' object location and 55% for 'correct' location.

The thresholds on the matching cost $\tau_{\mathrm{M}} = 20$, the neighborhood distance $\tau_{\mathrm{N}} = 50$, and an 11×1 block matching window for Eq. (3), were chosen empirically during algorithm development. No adjustment was required in testing. Further experiments (not detailed here) show that the method is quite robust to variation in these parameters.

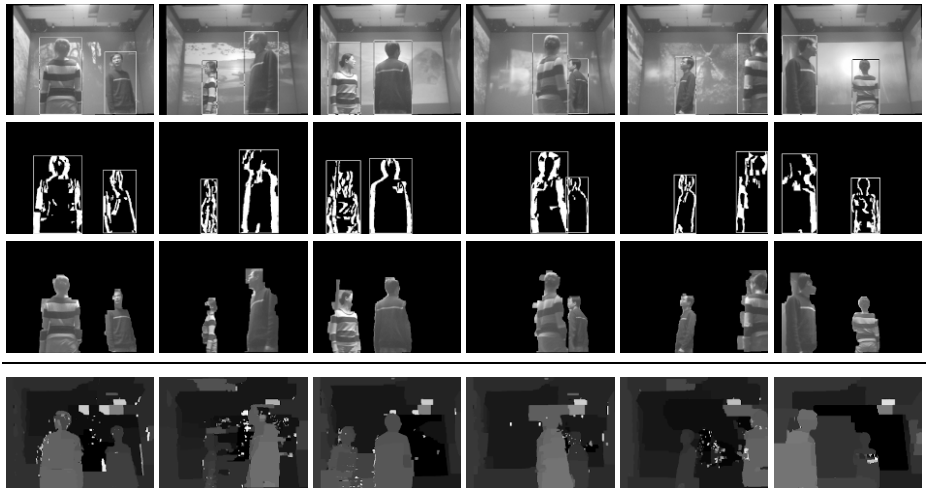
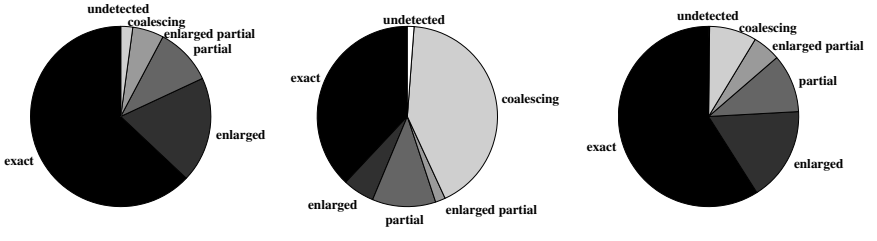


Fig. 10. Samples of multi-object segmentation in the presence of fast lighting and texture changes. Top row: original images after undistortion and rectification, with bounding boxes indicating segmentation results. Second row: contour grouping results. Third row: objects segmented from the scene. Last row: comparison with scene disparity map by graph cut [9].

Table 1. Object location accuracy with respect to the number of total objects

		no occlusion		**partial occl'n**		**all objects**	
total true objects		1767	100.00%	334	100.00%	2101	100.00%
accurate	exact bounding box	1113	62.99%	127	38.02%	1240	59.02%
inaccurate	enlarged bounding box	335	18.96%	19	5.69%	354	16.85%
	partial object	182	10.30%	38	11.38%	220	10.47%
incorrect	enlarged partial object	99	5.60%	6	1.80%	105	5.00%
	coalesced object	38	2.15%	140	41.92%	178	8.47%
	object undetected	0	0.00%	4	1.20%	4	0.19%
	false object					294	13.99%



(a) no occlusion (b) partial occlusion (c) all objects

Fig. 11. Object location accuracy with respect to the number of total objects

Fig. 10 also compares our results to those of the graph cut algorithm [11], one of the best stereo algorithms to date. Although graph cut produces acceptable scene disparity maps, its weakness on textureless regions, common in projected background, introduces many imperfections. Object segmentation based on this result would be challenging, requiring a large amount of post-processing.

Our unoptimized research implementation processes 640×480 monochrome image pairs at a rate of 3.8Hz (compared to graph cut's 0.0023Hz) on a 1.8GHz 32 bit AMD processor. Our analysis suggests that an improved implementation can be structured to achieve performance comparable to only a few linear passes over the input data. Crucially, of course, the construction of the BDM is offline and does not contribute to the online processing time.

Although overall performance of the system is encouraging, some problems remain, due to both external errors and algorithm issues.

The foremost source of external error is imprecise environment calibration. Inaccuracies in the background model BDM introduce systematic noise that confuses the segmenter and causes false objects. Using a special measurement device such as a laser pointer is expected to solve this problem. Other external errors such as camera synchronization error and video deinterlacing artifacts, whose effects are amplified by image undistortion and rectification, could be eliminated by employing progressive scan video cameras that take clock inputs.

Issues related to the algorithm itself include misleading contour grouping direction arising from texturelessness in both background and foreground, partial occlusion, the sensitivity of block matching to differences in camera response, viewing angle and specular lighting, and the dependence of boundary detection upon local intensity difference between background and foreground. However, based on the high success rate already achieved, exploiting the temporal coherence in an image sequence and adopting a higher-level tracker to propagate good segmentation results holds promise in all these areas.

Finally, the nature of the horizontally positioned stereo system results in a failure to detect horizontal or near horizontal object contours, such as at the top of the head and on the shoulders, and we have yet to investigate performance on highly textured foreground objects such as clothes with strong vertical patterns. However, adding vertical stereo into the framework and combining results on both axes can be expected to resolve both these concerns.

5 Conclusions

Disparity contours, an easily computed 2.5D stereo representation, is presented. On this basis, a new stereo image segmentation method to isolate and distinguish multiple foreground objects in a scene with fast illumination and texture change is developed. Without requiring full stereo reconstruction or tedious empirical parameter tuning, the method achieves near-real-time performance in software and generates not only the 2D image locations of objects but also boundaries, disparity and depth information, providing a natural extension to 3D processing. As no assumption is made on the shapes and textures of objects and environment, the proposed approach suits generic object segmentation tasks.

Acknowledgements. The authors thank Jianfeng Yin for his help with room geometry measurement and video acquisition and Jeremy R. Cooperstock for providing essential research facilities.

References

1. Ayer, S., Sawhney, H.S.: Layered representation of motion video using robust maximum-likelihood estimation of mixture models and MDL encoding. In: Int'l Conf. on Computer Vision, pp. 777–784 (1995)
2. Cucchiara, R., Grana, C., Piccardi, M., Prati, A.: Detecting moving objects, ghosts and shadows in video streams. IEEE Trans. Pattern Analysis and Machine Intelligence 25(10), 1337–1342 (2003)
3. Elder, J.H., Goldberg, R.M.: Ecological statistics of Gestalt laws for the perceptual organization of contours. Journal of Vision 2, 324–353 (2002)
4. Elder, J.H., Krupnik, A., Johnston, L.A.: Contour grouping with prior models. IEEE Trans. Pattern Analysis and Machine Intelligence 25(25), 1–14 (2003)
5. Foley, J.D., van Dam, A., Feiner, S.K., Hughes, J.F.: Computer Graphics: Principles and Practice in C, 2nd edn. Addison-Wesley, Reading (1997)
6. Fusiello, A., Trucco, E., Verri, A.: A compact algorithm for rectification of stereo pairs. Machine Vision and Applications 12(1), 16–22 (2000)
7. Ivanov, Y., Bobick, A., Liu, J.: Fast lighting independent background subtraction. Int'l Journal of Computer Vision 37(2), 199–207 (2000)
8. Jepson, A.D., Black, M.J.: Mixture models for optical flow computation. In: Computer Vision and Pattern Recognition, pp. 760–761 (1993)
9. Kolmogorov, V.: Software (2001–2003), `http://www.adastral.ucl.ac.uk/~vladkolm/software.html`
10. Kolmogorov, V., Criminisi, A., Blake, A., Cross, G., Rother, C.: Bi-layer segmentation of binocular stereo video. In: Computer Vision and Pattern Recognition, pp. 407–414 (2005)
11. Kolmogorov, V., Zabih, R.: Multi-camera scene reconstruction via graph cuts. In: Tistarelli, M., Bigun, J., Jain, A.K. (eds.) ECCV 2002. LNCS, vol. 2359, pp. 82–96. Springer, Heidelberg (2002)
12. Lin, M.H., Tomasi, C.: Surfaces with occlusions from layered stereo. IEEE Trans. Pattern Analysis and Machine Intelligence 26(8), 1073–1078 (2004)
13. Narayanan, P.J., Rander, P.W., Kanade, T.: Constructing virtual worlds using dense stereo. In: Int'l Conf. on Computer Vision, pp. 3–10 (1998)
14. Oliver, N.M., Rosario, B., Pentland, A.P.: A Bayesian computer vision system for modeling human interactions. IEEE Trans. Pattern Analysis and Machine Intelligence 22(8), 831–843 (2000)
15. Rittscher, J., Kato, J., Joga, S., Blake, A.: A probabilistic background model for tracking. In: Vernon, D. (ed.) ECCV 2000. LNCS, vol. 1843, pp. 336–350. Springer, Heidelberg (2000)
16. Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. In: Computer Vision and Pattern Recognition, pp. 246–252 (1999)
17. Sun, J., Zheng, N.N., Shum, H.Y.: Stereo matching using belief propagation. IEEE Trans. Pattern Analysis and Machine Intelligence 25(7), 787–800 (2003)
18. Sun, W., Cooperstock, J.R.: An empirical evaluation of factors influencing camera calibration accuracy using three publicly available techniques. Machine Vision and Applications 17(1), 51–67 (2006)
19. Sun, W., Spackman, S.P.: Multi-object segmentation by stereo mismatch. Machine Vision and Applications (2008), doi:10.1007/s00138-008-0127-1

20. Torr, P.H., Szeliski, R., Anandan, P.: An integrated Bayesian approach to layer extraction from image sequences. IEEE Trans. Pattern Analysis and Machine Intelligence 23(3), 297–303 (2001)
21. Toyama, K., Krumm, J., Brumitt, B., Meyers, B.: Wallflower: principles and practice of background maintenance. In: Int'l Conf. on Computer Vision, pp. 255–261 (1999)
22. Wang, J.Y., Adelson, E.H.: Layered representation for motion analysis. In: Computer Vision and Pattern Recognition, pp. 361–366 (1993)
23. Weiss, Y., Adelson, E.H.: A unified mixture framework for motion segmentation: incorporating spatial coherence and estimating the number of models. In: Computer Vision and Pattern Recognition, pp. 321–326 (1996)
24. Wren, C.R., Azarbayejani, A.J., Darrell, T.J., Pentland, A.P.: Pfinder: real-time tracking of the human body. IEEE Trans. Pattern Analysis and Machine Intelligence 19(7), 780–785 (1997)
25. Zhang, L., Curless, B., Seitz, S.M.: Rapid shape acquisition using color structured light and multi-pass dynamic programming. In: Int'l Symposium on 3D Data Processing Visualization and Transmission, pp. 24–36 (2002)

Video-Based Camera Tracking Using Rotation-Discriminative Template Matching

David Marimon and Touradj Ebrahimi

Multimedia Signal Processing Group
Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland
david.marimon@a3.epfl.ch, touradj.ebrahimi@epfl.ch

Abstract. This paper presents a video-based camera tracker that combines marker-based and feature point-based cues in a particle filter framework. The framework relies on their complementary performance. Marker-based trackers can robustly recover camera position and orientation when a reference (marker) is available, but fail once the reference becomes unavailable. On the other hand, feature point tracking can still provide estimates given a limited number of feature points. However, these tend to drift and usually fail to recover when the reference reappears. Therefore, we propose a combination where the estimate of the filter is updated from the individual measurements of each cue. More precisely, the marker-based cue is selected when the marker is available whereas the feature point-based cue is selected otherwise. Feature points are dynamically found in scene and used for further tracking. Evaluations on real cases show that the fusion of these two approaches outperforms the individual tracking results. A critical aspect of the feature point-based cue is to robustly recognise the feature points depite rotations of the camera. A novelty of the proposed framework is the use of a rotation-discriminative method to match feature points.

1 Introduction

Combination of tracking techniques has proven to be necessary for some camera tracking applications. To reach a synergy, techniques with complementary performance have first to be identified. Research on camera tracking has concentrated on combining sensors within different modalities (e.g. inertial, acoustic, optic). However, this identification is possible within a single modality: video trackers. Video-based camera tracking can be classified into two categories that have compensated weaknesses and strengths: bottom-up and top-down approaches [1]. For the first category, the six Degrees of Freedom (DoF), 3D position and 3D orientation, estimates are obtained from low-level 2D features and their 3D geometric relation (such as homography, epipolar geometry, CAD models or patterns), whereas for the second group, the 6D estimate is obtained from top-down state space approaches using motion models and prediction. *Marker-based systems* [2] can be classified in the first group. Although they have a high detection rate and estimation speed, they still lack tracking robustness: the marker(s) must be always visible thus limiting the user actions. In contrast to bottom-up approaches, top-down techniques such as *filter-based camera tracking* allow track continuation when the reference is temporarily unavailable (e.g. due to occlusions). They use predictive motion

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 232–243, 2008.

models and update them when the reference is again visible [3,4]. Their weakness is, in general, the drift during the absence of a stable reference (usually due to features difficult to recognise after perspective distortions). Filter-based camera tracking generally uses available data such as feature points to correct the filtered state. The problem with feature points is to reliably recognise them. Most techniques use descriptors based on the grey-level or colour histogram or directly the intensity (templates) of their neighbourhood [3,4]. Feature points change their appearance at consecutive frames due to camera motion. Therefore, methods that robustly recognise feature points despite those changes have to be employed.

In this paper, we present a particle-filter based camera tracker. The main purpose of this framework is to take advantage of the complementary performance of two particular video-trackers. The system combines the measurements of a marker-based cue (MC) and a feature point-based cue (FPC). The MC tracks a square marker using its contour lines. The FPC tracks the feature points found in the scene. The proposed framework extends the camera traking system presented in [5]. In this previous work, only the corners of the marker are used and the method to recognise feature points is very sensible to rotations of the camera. We propose a novel use of the rotation discriminative template matching (RDTM) method described in [6]. More precisely, this method is employed here to recognise feature points despite large rotations.

The paper is structured as follows. Section 2 describes similar works. The techniques involved in the combination and the proposed tracker are presented in Section 3. Several experiments and results are given in Section 4. Conclusions and future research directions are finally discussed.

2 Related Work

In hybrid tracking, systems that combine diverse tracking techniques have shown that the fusion obtained enhances the overall performance [7].

The commonly developed fusions are inertial-acoustic and inertial-video [7]. Inertial sensors usually achieve better performance for fast motion. On the other hand, in order to compensate for drift, an accurate tracker is needed for periodical correction. The advantage of using a bottom-up approach such as a marker-based tracker is that drift is automatically reduced each time the detection occurs. Several works have combined marker-based approaches with inertial sensors [8,9]. [9] presented a square marker-based tracker that fuses its data with an inertial tracker, in a Kalman filtering framework. Among the existing marker-based trackers, two recent works, [10] and [11] stand out for their robustness to illumination changes and partial occlusions. [10] takes advantage of machine learning techniques, and trains a classifier with a set of markers under different conditions of light and viewpoint. No particular attention is given to occlusion handling. [11] uses spatial derivatives of grey-scale image to detect edges, produce line segments and further link them into squares. This linking method permits the localisation of markers even when the illumination is different from one edge to the other. The drawback of this method is that markers can only be occluded up to a certain degree. More precisely, the edges must be visible enough to produce straight lines that cross at the corners.

However, little attention has been given to fusing diverse techniques from the same modality. Several researchers have identified the potential of video-based tracking fusion [1,12]. Among these, [1] is the only reported work to fuse data from a single camera. Their system switches between a model-based tracker and a feature point-based tracker, similar to that of [4]. Nonetheless, this framework takes limited advantage of the filtering framework and still needs the assistance of an inertial sensor.

Recent works have addressed the problem of robustly identifying feature points in camera tracking frameworks [13,15]. In both cases, the application of invariant descriptors for correct feature point matching has brought important improvements. Sim *et al.* [13] use SIFT features [14], which have high scale and rotation invariance enabling accurate tracking. However, the extraction and description of SIFT features makes the mapping of the scene more complicated. Indeed, the data association of feature points between frames cannot be used in a straightforward manner because the descriptors are scale invariant and hence the features have many different scales. Therefore, the association is done by traversing all the list of feature descriptors. This process has a large computational cost and the overall system runs at 11.9 seconds per frame. Chekhlov *et al.* [15] propose a multi-resolution descriptor based also on SIFT. The approach differs from [13] in that the extraction of feature points is done at a fixed scale. In order to be scale invariant, several SIFT descriptors at different scales are stored for each feature. At runtime, the scale is selected according to camera pose and 3D feature position. Once the scale is selected, the validation can be computed.

Those descriptors differ from the descriptor presented in [6] mainly in the fact that rotation information is lost. We propose to exploit this information during the filter update by associating it to the estimated camera rotation.

3 System Description

This section describes the parameters of the filter, how the marker-based and the feature point-based cues are obtained, as well as the procedure used to fuse them in the filter.

3.1 Particle Filter Equations

We target applications where the camera is hand-held or attached to the user's head. Under these circumstances, Kalman filter-based approaches although extensively used for ego motion tracking, lead to a non optimal solution because the motion is not white nor has Gaussian statistics [16]. To avoid the Gaussianity assumption, we have chosen a camera tracking algorithm that uses a particle filter. More precisely, we have chosen a sample importance resampling (SIR) filter. For more details on particle filters, the reader is referred to [17].

Each particle n in the filter represents a possible camera pose

$$T_n = [t_X, t_Y, t_Z, \mathrm{rot}_W, \mathrm{rot}_X, \mathrm{rot}_Y, \mathrm{rot}_Z]_n, \tag{1}$$

where t are the translations and rot is the quaternion for the rotation. T determines the 3D relation of the camera with respect to the world coordinate system. We have avoided

adding the velocity terms so as not to overload the particle filter (which would otherwise affect the speed of the system).

For each video frame, the filter follows two steps: prediction and update. The probabilistic motion model for the prediction step is defined as follows. The process noise (also known as transition prior $p(T_n(k)|T_n(k-1))$) is modelled with a Uniform distribution centred at the previous state $T_n(k-1)$ (frame $k-1$), with variance q (process noise's -also called system noise- vector of hyper-parameters). The reason for this type of random walk motion model is to avoid any assumption on the direction of the motion. This distribution enables faster reactivity to abrupt changes. The propagation for the translation vector is

$$T_n(k)\big|_{t_X,t_Y,t_Z} = T_n(k-1)\big|_{t_X,t_Y,t_Z} + u_t \tag{2}$$

where u_t is a random variable coming from the uniform distribution, particularised for each translation axis. The propagation for the rotation is

$$T_n(k)\big|_{rot} = u_{rot} \times T_n(k-1)\big|_{rot} \tag{3}$$

where $\times$ is a quaternion multiplication and u_{rot} is a quaternion coming from the uniform distribution of the rotation components. In the update step, the weight of each particle n is calculated using its measurement noise (likelihood)

$$w_n = p(Y|T_n), \tag{4}$$

where w_n is the weight of particle n and Y is the measurement. The key role of the combination filter is to switch between two sorts of likelihood depending on the type of measurement that is used: MC or FPC. Once the weights are obtained, these are normalised and the update step of the filter is concluded. The corrected mean state $\widehat{T}$ is given by the weighted sum of T_n. $\widehat{T}$ is used as output of the camera tracking system.

3.2 Marker-Based Cue (MC)

We use the marker-based system provided by [18] to calculate the transformation T between the world coordinate frame and that of the camera (3D position and 3D orientation). As explained in Section 3.4, this transformation is the measurement fed into the filter for update.

At each frame, the algorithm searches for a square marker (see Figure 1) inside the field-of-view (FoV).

If a marker is detected, the transformation can be computed. The detection process works as follows. First, the frame is converted to a binary image and the black marker contour is identified. If this identification is positive, the 6D pose of the marker relative to the camera (T) is calculated. This computation uses only the geometric relation of the four projected lines that contour the marker in addition to the recognition of a non-symmetric pattern inside the marker [18]. When this information is not available, no pose can be calculated. This occurs in the following cases: markers are partially or completely occluded by an object; markers are partially or completely out of the FoV; or not all lines can be detected (e.g., due to low contrast).

Fig. 1. Square marker used for the MC

3.3 Feature Point-Based Cue (FPC)

In order to constrain the camera pose estimation, the back-projection of feature points in the scene can be used. For this purpose, both the 3D location of the feature point P and the 2D back-projection p is needed. In homogeneous coordinates,

$$p = K \cdot [R|t] \cdot P, \tag{5}$$

where K is the calibration matrix (computed off-line), R is the rotation matrix formed using the quaternion rot and $t = [t_X, t_Y, t_Z]^T$ is the translation vector.

Natural feature points in unprepared environments appear in objects at unknown locations. Hence, the 3D location of feature points in the world coordinate frame is generally unavailable. However, the combination framework proposed here admits a certain preparation of the environment, this is, a marker is available. Since the world coordinate frame is fixed to the marker and the real size of the marker is known, the 3D location of any point in the marker is known. We take advantage of this fact and propose to use the corners as feature points in the scene.

Although we have proved in our previous work that these points provide a reliable measurement for camera tracking [5], they might not always be available. For instance, because a corner is occluded by an object or it is outside of the FoV. In this case, it is interesting to have other feature points to rely on. As explained before, in order to constrain the camera pose, the 3D position of a feature point must be available. However, the inverse procedure can also be done. Indeed, from Equation (5) one deduces that the 3D world coordinates of a point can be computed if the camera pose $[\mathbf{R}|\mathbf{t}]$ is known. Since the filter keeps an estimate of this pose, it is possible to calculate the 3D position of feature points. Once this location is computed, a new feature point can be added to the map of feature points that constrain the camera pose. This process is detailed in [19].

The intensity level and gradient information are chosen as a description of the feature points, for further recognition. Each time a feature point is added to the map the template of its neighbourhood is stored. At this time, rotated versions of this template are generated. The orientation gradient is computed for each of these versions and the information is summarised in a single robust orientation histogram. The final descriptor of a feature point is composed by the histogram and the rotated templates. The amount of rotated versions is proportional to the number of bins in the histogram.

At runtime, the feature points in the map are searched in the video frame. A region is defined around the estimated location of each feature point. Assume, for the moment, that those regions are known. Each region is matched with the corresponding descriptor.

The result of this matching is a correlation score together with a bin-wise estimated rotation, for each pixel inside the region. More precisely, the result indicates which rotated version $\Theta(x, y)$ of the template gives the highest correlation $\Psi(x, y)$ at each pixel (x, y). Further details about the descriptor and the RDTM process can be found in [6].

As explained in the next section, the set of correlation scores and estimated rotations is the measurement fed into the filter for update. Each feature point that is positively matched makes the filter converge to a more stable estimate. Three points are necessary to robustly determine the six DoF. However, the filter can be updated even with only one feature point. A reliable feature point might be unavailable in the following situations: a point is occluded by an object; a point is outside of the FoV; the region does not contain the feature point (due to a bad region estimation); or the point is inside the region but no correlation is beyond the threshold (e.g., because the viewpoint is drastically changed).

3.4 Cues Combination

The goal of the system is to obtain a synergy by combining both cues. Individual weaknesses previously described are thus lessened by this combination. Special attention is given to the occlusion and illumination problems in the MC and the drift in the FPC.

At initialisation, the value of all particles of the filter is set to the transformation estimated by the marker-based cue T_{MC}.

As long as the the marker is detected, the system uses the MC measurement to update the particle filter (Y =MC). The likelihood is modelled with a Cauchy distribution centered at the measurement T_{MC}

$$p(T_{MC}|T_n) = \prod_i \frac{r_i}{\pi \cdot ((T_{n,i} - T_{MC,i})^2 + r_i^2)}, \tag{6}$$

where r is the measurement noise and i indexes the elements of the vectors. This particular distribution's choice has its origin in the following reasoning. In the resampling step of the filter, particles with insignificant weights are discarded. A problem may arise when the particles lie on the tail of the measurement noise distribution. The transition prior $p(T_n(k)|T_n(k-1))$ determines the region in the state-space where the particles fall before their weighting. Hence, it is relevant to evaluate the overlap between the likelihood distribution and the transition prior distribution. When the overlap is small, the number of particles effectively resampled is too small. Figure 2 shows an instance of overlapping region. It must be pointed out that due to computing limits, some values fall to zero even though their real mathematical value is greater than that (the support of a Gaussian distribution is the entire real line). In the example of this figure, there is no sufficient computed overlap for the Gaussian distribution (commonly used), whereas the tail of the Cauchy distribution covers the necessary state-space. Therefore, we have chosen a long-tailed density that better covers the state-space, while still being a realistic measurement noise [20].

On the other hand, when the MC fails to detect the marker, the system relies on the FPC (Y =FPC) and another likelihood is used. As a previous step to looking for the new location of the feature points (see Section 3.3), it is necessary to calculate the *regions* around the estimated location of each feature point. For each feature point, all

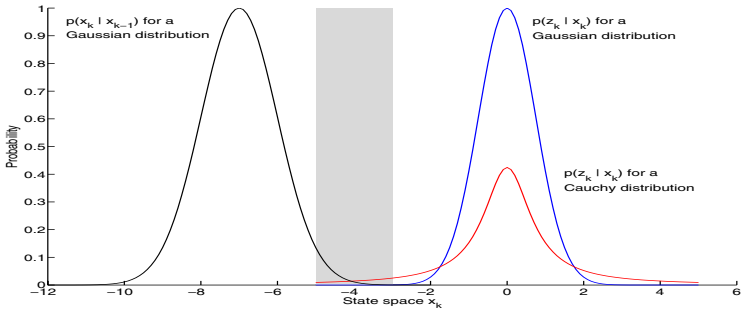


Fig. 2. Overlap between transition prior distribution and the likelihood distribution: modelled with a Gaussian (no overlap) and with a Cauchy distribution (thick line)

the back-projections given the transformations T_n are computed (see Eq. 5). The *region* is the bounding box containing all these back-projections. These bounding boxes are fed into the FPC and the matching results are obtained in return. The weights can then be calculated. First, a set of 2D coordinates is obtained by thresholding $\Psi(x, y)$.

$$S_j = \left\{ [c_x, c_y] \middle| \Psi_j(c_x, c_y) > th_{corr} \right\}, \tag{7}$$

where j indexes the feature points mapped from the scene. Second, for each particle, a subset is kept with the points in S_j that are within a certain Euclidean distance from the corresponding back-projection $[p_{n,x}, p_{n,y}]$

$$S_{n,j} = \left\{ [c_x, c_y] \in S_j \middle| dist(c, p_n) < th_{dist} \right\}. \tag{8}$$

Finally, the weight is computed. The weight of the particle n is proportional to the correlation Ψ_j achieved in the subsets $S_{n,j}$. Furthermore, this is refined with the orientation Θ_j estimated by the RDTM process. This orientation should have a rough correspondence with the rotation of the camera about the Z axis. The more perpendicular is the original template to the current pose of the camera, the higher the chances of the estimated orientation being similar to the rotation about the Z axis. We take advantage of this fact. Indeed, the weights are forced to be proportional also to the difference between the orientation Θ_j and the rotation around the Z axis of the corresponding particles's state $\psi_{Z,n}$

$$w_n = exp\left(\sum_{j=1}^{L} \sum_{[x,y] \in S_{n,j}} \Psi_j(x, y) \cdot exp - \left(\frac{(\psi_{Z,n} - \hat{\psi}_{Z,j}) - \Theta_j(x, y) \cdot \Delta}{\alpha \cdot \Delta} \right)^2 \right), \tag{9}$$

where L is the number of feature points, $\Delta = 360/N$ is the quantisation step of the orientation according to the number of bins N (see Section 3.3), $\hat{\psi}_{Z,j}$ is the rotation of the camera at the initialisation of the feature point, and α is a tunable parameter. Weighting the particles according to the correlation gives already a strong validation for the data association between feature points and the point in the image plane where they lie. Reinforcing this validation with the orientation permits to avoid confusion with

points with high correlation but unexpected orientation according to the camera's pose. Therefore, α can be tuned to vary this reinforcement of the data association. In our case, this parameter is fixed to a high value ($\alpha = N/2$) as the perpendicularity of the camera with respect to the template of a feature point cannot be assured a priori. It is also possible to make this parameter vary according to the angle of rotation in X and Y axes, for instance $\alpha \propto \sum |\psi_{X,n} - \hat{\psi}_{X,j}| + |\psi_{Y,n} - \hat{\psi}_{Y,j}|$. This option is not considered for simplicity purposes.

As it can be seen, the likelihood for the FPC measurement is much less straightforward to compute than the MC. Nevertheless, the weights can be calculated independently of the number of feature points recognised whereas the likelihood for the MC is available only if the marker is visible.

Algorithm 1 expresses the process followed by the combination. It is assumed that the filter has been initialised at the first detection of the marker. The description of the marker is stored in the *pattern* variable.

Algorithm 1. Combination procedure.

```
loop
  vframe ← getVideoFrame()
  marker ← detectMarker( vframe )
  if pattern.correspondsTo( marker ) then
    T_MC ← MC.calcTransformation( marker )
    T̂ ← filter.updateFromMC( T_MC )
  else
    reg ← filter.calcRegions()
    for j = 1 to NumberOfFeaturePoints do
      [Θ_j, Ψ_j] ← RDTM( reg , vframe, descriptors_j )
    end for
    T̂ ← filter.updateFromFPC( Θ_{j=1...L} , Ψ_{j=1...L} )
  end if
  filter.findNewFeaturePoints( vframe )
end loop
```

This filtering framework has several advantages. Combination through a filter provides a continuous estimate which is free of jumps that disturb the user's interaction. Frameworks often fall into static solutions giving little opportunity for shaping. The likelihood switching method proposed is generic enough to be used with very different types of cues or sensors such as inertial.

4 Experiments

In order to assess the performance of the camera tracking system, we have performed several experiments. Two sequences are used. The first one is generated synthetically. The second one is recorded with a hand-held camera. For the first one, the ground truth is known whereas for the second one a qualitative measure is used. When the camera position with respect to the world coordinate frame is known, it is possible to

add virtual objects at a 3D position in the world coordinate space. This is generally known as Augmented Reality. If the alignment between a virtual object and the real scene is fixed, the object should move accordingly to the cameras motion as if it was placed in the real world. A qualitative measure is found by observing how static a fixed virtual object is with respect to the real world.

4.1 Evaluation of the Combination of Cues

An experiment is conducted to analyse the tracking performance in front of occlusions of the marker. As stated before, one of our goals is to cope with the loss of track of the MC when the marker is occluded. In our framework, tracking can continue by using the FPC. Two techniques are compared in this case. On the one hand, ARToolkit [18], which is equivalent to use the MC alone. On the other hand, our framework combining MC and FPC.

Snapshots from several frames of the augmented sequence are shown in Fig. 3.

(a) Snapshots of a manual occlusion.

(b) Snapshots of a manual occlusion.

(c) Snapshots while the marker is escaping the field of view.

Fig. 3. Experiment with occlusions. A virtual teapot is placed on the marker to show correct alignment. When the teapot is red, the framework uses the MC, whereas when it is green, the framework relies on the FPC.

4.2 Evaluation of the RDTM for Camera Tracking

In [6], the RDTM method to recognise regions is described. This method is tailored to match a template despite of a 2D rotation, as well as detect the rotation that the template has undergone. In this paper, experiments have shown the accuracy of the method on several images rotated over the perpendicular axis (2D rotations). We want to evaluate here the improvement brought by the RDTMwhen compared to a simpler but commonly used method [3,4,5].

Two feature point-based camera trackers with different matching techniques are compared. In the first one, the recognition is performed with the Normalised Cross-Correlation (NCC) of the templates. In the second one, matching of feature points

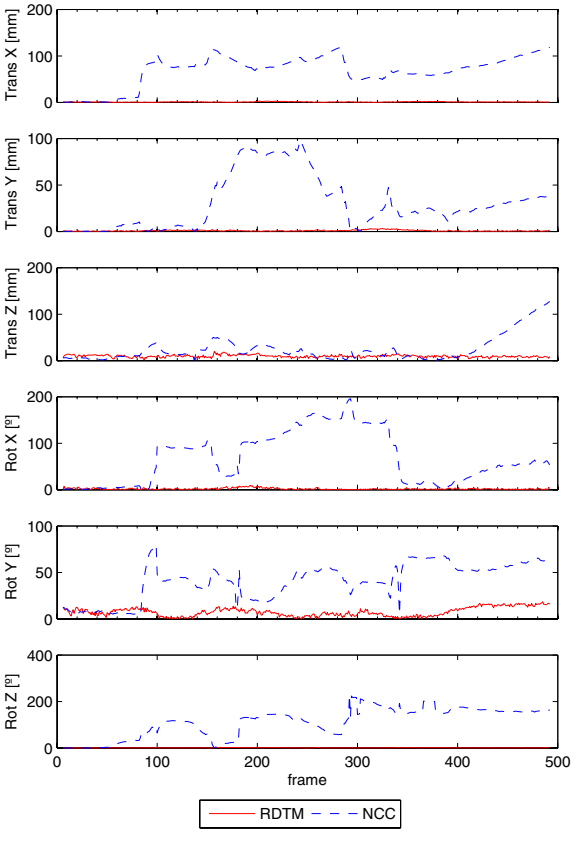


Fig. 4. Experiment with different feature point recognition methods. Comparison between NCC and RDTM. Absolute error of the translation and rotation in X,Y and Z axes (Renens sequence).

is done with our RDTM method. The experiment is conducted with the synthetic sequence.

Fig. 4 shows one instance of the absolute error of each axis of the compared techniques.

Matching with NCC fails as soon as a large rotation around the Z axis occurs (around frame 50). As a consequence, this tracker looses all references and starts to drift. On the other hand, the rotation-discriminative method allows a continuous track of the feature points and hence accurate camera pose estimation. Indeed, the Root Mean Square Error achieved for the Z axis is very low: 0.79 degrees.

5 Conclusions

We have presented a combination of video-based camera trackers within a particle filter framework. The filter uses two cues provided by a marker-based approach and a

feature point-based one. We introduce a novel use of a rotation-discriminative template matching (RDTM) method for camera tracking.

Experiments show that the proposed combination produces a synergy. In particular we have shown robustness in front of occlusions of the marker. Moreover, we have demonstrated the convenience of using the RDTM by comparison to other commonly used template matching.

In our future research, we will focus on extending the application of the RDTM to scale invariance by exploiting the knowledge of the estimated distance between the camera and the feature points.

References

1. Okuma, T., Kurata, T., Sakaue, K.: Fiducial-less 3-D object tracking in AR systems based on the integration of top-down and bottom-up approaches and automatic database addition. In: Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR), p. 260 (2003)
2. Zhang, X., Fronz, S., Navab, N.: Visual marker detection and decoding in AR systems: A comparative study. In: Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR), September–October 2002, pp. 97–106 (2002)
3. Davison, A.: Real-time simultaneous localisation and mapping with a single camera. In: Proc. Intl. Conf. on Computer Vision (ICCV) (2003)
4. Pupilli, M., Calway, A.: Real-time camera tracking using a particle filter. In: Proc. British Machine Vision Conference (BMVC), pp. 519–528 (September 2005)
5. Marimon, D., Ebrahimi, T.: Combination of video-based camera trackers using a dynamically adapted particle filter. In: 2nd Intl. Conf. on Computer Vision Theory and Applications (VISAPP 2007) (2007)
6. Marimon, D., Ebrahimi, T.: Efficient rotation-discriminative template matching. In: Rueda, L., Mery, D., Kittler, J. (eds.) CIARP 2007. LNCS, vol. 4756, pp. 221–230. Springer, Heidelberg (2007)
7. Allen, B., Bishop, G., Welch, G.: Tracking: Beyond 15 minutes of thought. In: Course Notes, Ann. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH) (2001)
8. Kanbara, M., Fujii, H., Takemura, H., Yokoya, N.: A stereo vision-based augmented reality system with an inertial sensor. In: Proc. IEEE and ACM Intl. Symp. on Augmented Reality (ISAR), pp. 97–100 (October 2000)
9. You, S., Neumann, U.: Fusion of vision and gyro tracking for robust augmented reality registration. In: Proc. IEEE Virtual Reality (VR), pp. 71–78 (2001)
10. Claus, D., Fitzgibbon, A.: Reliable fiducial detection in natural scenes. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3024, pp. 469–480. Springer, Heidelberg (2004)
11. Fiala, M.: ARTag, a fiducial marker system using digital techniques. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Washington, DC, USA, vol. 2, pp. 590–596. IEEE Computer Society, Los Alamitos (2005)
12. Satoh, K., Uchiyama, S., Yamamoto, H., Tamura, H.: Robot vision-based registration utilizing bird's-eye view with user's view. In: Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR), pp. 46–55 (October 2003)
13. Sim, R., Elinas, P., Griffin, M., Little, J.J.: Vision-based SLAM using the Rao-Blackwellised particle filter. In: Proc. IJCAI Workshop on Reasoning with Uncertainty in Robotics (RUR), Edinburgh, Scotland, pp. 9–16 (2005)
14. Lowe, D.: Distinctive image features from scale-invariant keypoints. Intl. Journal of Computer Vision 60(2), 91–110 (2004)

15. Chekhlov, D., Pupilli, M., Mayol-Cuevas, W., Calway, A.: Real-time and robust monocular slam using predictive multi-resolution descriptors. In: 2nd International Symposium on Visual Computing (November 2006)
16. Chai, L., Nguyen, K., Hoff, B., Vincent, T.: An adaptive estimator for registration in augmented reality. In: Proc. IEEE and ACM Intl. Workshop on Augmented Reality (IWAR), pp. 23–32 (1999)
17. Arulampalam, M., Maskell, S., Gordon, N., Clapp, T.: A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. IEEE Trans. on Signal Processing 50(2), 174–188 (2002)
18. Kato, H., Billinghurst, M.: Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In: Proc. Intl. Workshop on Augmented Reality (IWAR), pp. 85–94 (October 1999)
19. Marimon, D.: Advances in top-down and bottom-up approaches to video-based camera tracking. PhD thesis, EPFL (2007)
20. Ichimura, N.: Stochastic filtering for motion trajectory in image sequences using a monte carlo filter with estimation of hyper-parameters. In: Proc. Intl. Conf. on Pattern Recognition (ICPR), vol. 4, pp. 68–73 (2002)

Energy Association Filter for Online Data Association with Missing Data

El Abed Abir, Dubuisson Séverine, and Béréziat Dominique

Laboratoire d'Informatique de Paris 6, Université Pierre et Marie Curie
104 avenue du Président Kennedy, 75016 Paris, France
abir.elabed@lip6.fr

Abstract. Data association problem is of crucial importance to improve online object tracking performance in many difficult visual environments. Usually, association effectiveness is based on prior information and observation category. However, some problems can arise when objects are quite similar. Therefore, neither the color nor the shape could be helpful informations to achieve the task of data association. Likewise, a problem can also arise when tracking deformable objects, under the constraint of missing data, with complex motions. Such restriction, *i.e.* the lack in prior information, limit the association performance. To remedy, we propose a novel method for data association, inspired from the evolution of the object dynamic model, and based on a global minimization of an energy. The main idea is to measure the absolute geometric accuracy between features. Parameterless constitutes the main advantage of our energy minimization approach. Only one information, the position, is used as input to our algorithm. We have tested our approach on several sequences to show its effectiveness.

Keywords: Online data association, energy minimization, prior informationless and non-rigid motion.

1 Introduction

Traditionally, multiple object tracking deals with the state estimation of moving objects. A track is a state trajectory estimated from the available measurements that have been associated with the same object [2]. The main difficulty comes from the assignment of a given measurement to a object model. These assignments are generally unknown, as are the true object models. The idea of data association remains to find a partition of observations such that each element is generated by a object, or clutter, whose statistical properties differ from one object to another. The literature contains some classical approaches: we can distinguish the deterministic approaches from the probabilistic ones.

Deterministic approaches select the best of several candidate associations, without taking into account its context correctness, by using a score function [2]. The simplest deterministic method for data association is the Nearest-Neighbor Standard Filter (NNSF) [6] that selects the closest validate measurement to a predicted object and uses it for its state estimation. Usually, the distance measure used is the Mahalanobis one. Since the filter does not take into account the possibility of incorrect associations, the performance of this filter might be poor in some cases, resulting in an incorrect association between measurements and objects. In some tracking applications, the color is

J. Braz et al. (Eds.): VISIGRAPP 2007, CCIS 21, pp. 244–257, 2008.

exploited for the problem of data association. One can measure the color histogram difference between a measurement and the objects of the previous frame using the histogram intersection technique. Unfortunately, the color metric is not sufficient for a correct data association in many cases: for deformable objects, which color distribution may differ from one frame to another, or in case of several quite identical objects.

Probabilistic approaches are based on posterior probability and make an association decision using the probability error [3]. Among probabilistic approaches, we can cite the most general one, called Multiple Hypothesis Tracking (MHT) [2]. In MHT, the multiple hypotheses are formed and propagated, implying the calculation of every possible hypothesis. Due to the exploring of a high-dimensional joint state space, this method is computational intensive. Another strategy for multiple object tracking association is the Probability Data Association Filter (PDAF) [3], that assigns an association probability to each measurement and uses these probabilities to weight the measurements for track update. The original PDAF formulation has some limitations: it assumes that all measurements come from the track being updated, that is not true in case of dense object conditions. The Joint Probability Data Association (JPDA) [8] uses a weighted sum of all measurements near the predicted state, each weight corresponding to the posteriori probability for a measurement to come from a object. JPDAF provides an optimal data solution in the Bayesian framework. However, the number of possible hypothesis increases rapidly with the number of objects, requiring prohibitive amount of time calculating.

Generally, an effective data association method is based on prior information and observation category. Once we have a lack of prior information, that can happen when the observer has no information concerning the system, the association task becomes difficult. Such cases can occur when the observed system is deformable, moreover, when we observe with minor information about the movement, multiple objects that are quite similar even non distinguishable. It could be more complicated if we have a considerable interval of time between observations and where the observer has no prior information about object's motion. Likewise, if we only observe object position, we can meet the case where a measurement is equidistant from several objects: all object association probabilities are relatively the same and it is difficult to associate the good measurement with the good object. As far as, no association method can handle all the cases illustrated previously.

In this paper, we propose a novel method for data association based on minimization of an energy magnitude E and adapted to the circumstances described previously. This energy, inspired from object motion, measures the geometric accuracy between features and associates measurement y (given by sensor) with object k if $(E_k)_y$ is minimized. The main advantages of this energy are followed. It does not require parameters, does not need prior knowledge and does not a time-consumer. Exclusively one information about object is used: its position. Besides, it can handle the problem of association when a measurement falls within the validation regions for several objects and is equidistant from them.

The outline of this paper is as follows. In section 2, we expose the energy minimization approach, derive its geometrical representation and its mathematical model. The proposed method is then evaluated and tested on several sequences in section 3. Finally, concluding remarks and perspective works are given in section 4.

2 Energy Association Filter (EAF)

We first need to define some terms that will be often used in this paper. We dispose a video sequence describing a dynamical scene. It is observed by a set of sensors, each one can deliver exactly one observation at a precise time step t. Each observation contains at least one measurement: a position. The number of available measurements can differ from one observation to another. Each measurement can be associated with a specific object in the scene (*i.e.* object), or can be a false alarm. At a specific time t, observations are assumed to be available from N_{obs} sensors. The set of observations coming from all sensors is given by $y = (y^1, ..., y^{N_{\text{obs}}})$, where $y^i = (y^i_1, ..., y^i_{M^i})$ is the vector containing the M^i measurements coming from the i^{th} sensor, also called observation. We suppose that each sensor can generate at most one observation, containing at least one measurement at a particular time step and that the number of measurements delivered by the sensors varies with time. When an observation is available, our goal is to associate a maximum one measurement per object. The total number of objects is K.

2.1 Energy Minimization Modelling

Generally, an effective data association method is based on measurements category. When the measurement is limited to the position, and falls inside the validation region of several objects and is equidistant from them (see Figure 1.(a)), it will be associated with all these objects if we use the NNSF or Monte Carlo JPDAF approaches. As well as, in multiple object tracking, feature objects can be quite similar. Accordingly, even if information about their color distribution or shape is available, the association task is difficult under such assumptions or impossible in case of complex dynamics.

In this paper, we propose an algorithm for data association restricted to one category of measurement: the position. Furthermore, we affirm the total lack of prior information concerning objects: exclusively the two anterior predicted positions are used. We will first give the concept of our approach before starting its mathematical modeling. Our intention is to formalize a method able to associate a measurement according to the restrictions displayed in section 1. We define a novel energy E inspired from the object's dynamic evolution. The dynamic model is described in terms of displacements in the object space (x, y). If we only consider the linear translation in one direction, the problem of data association is limited to the computation of the Mahalanobis distance energy E^1 (see after for details). Thus, in case of complex dynamics such as non linear displacements, oscillatory motions and non-constant velocities, we are vis-a-vis a problem because E^1 will be an inadequate informative source. To remedy, we incorporate a second energy E^2 which measures the absolute accuracy between the dynamic features and indicates how much their parameters are close. Moreover, we distinguish some dynamic cases, that will be clarified by geometric descriptions afterward, where we need to compensate E^2 by the proximity energy evolution E^3 for a better association of te available data.

The energy E is only computed when the measurement falls within several validation regions. We consider a measurement as a clutter if it is not included in any validation region. In our case, the validation region is an ellipsoid that contains a given probability mass under the Gaussian assumption. The minor and major axes of this ellipsoid are

respectively given by the largest and smallest eigenvalues of the covariance matrix, their directions are given by the corresponding eigenvectors, and the center is the mean of the object.

We define the energy between the k^{th} object ($k = 1, \ldots, K$) and the measurement y_j^i, *i.e.* j^{th} measurement of the i^{th} sensor, by:

$$(E_k)_{y_j^i} = \frac{1}{\sqrt{3}} \sum_{l=1}^{3} \alpha_l (E_k^l)_{y_j^i} \tag{1}$$

where $\alpha_l = \frac{1}{\sum_{k=1}^{K} ||(E_k^l)_{y_j^i}||}$ is a weighted factor introduced to sensibly emphasize the relative importance attached to the energy quantities E^l.

Before interpreting each energy, we consider an object A and a measurement y_j^i. Besides, we call (see Figure 1 for illustration):

– $\hat{A}(t-2)$, $\hat{A}(t-1)$, $\hat{A}(t)$ and $\hat{A}_1(t+1)$: prediction of A at $t-2$, $t-1$, t and $t+1$ by using the initial dynamic model;

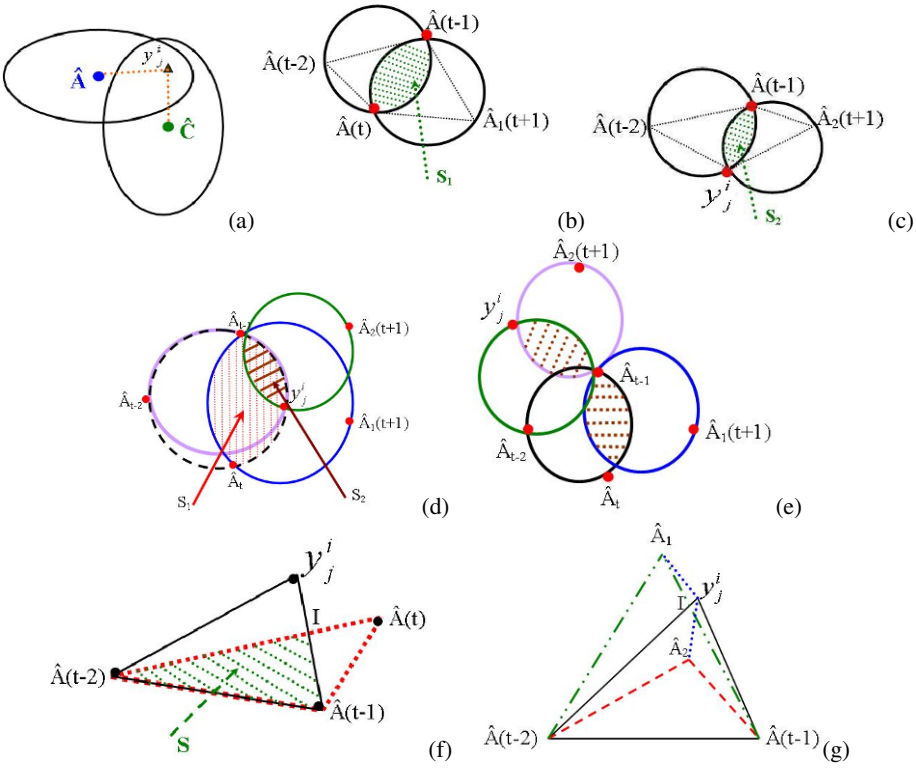


Fig. 1. (a) Measurement y_j^i falls inside the two validation regions of A and C; (b-c-f) Visualize the intersection surfaces $\{S_1, S_2, S\}$; (d-e) Project the difference between the sufaces S_1 and S_2 extracted from two dynamical model; (g) Shows the intersection surfaces where two predictions at instant t, $\hat{A}_1$ and $\hat{A}_2$, are equidistant from y_j^i

– $\hat{A}_2(t+1)$: prediction of A at $t+1$ by using the updated dynamic model. In this case, the measurement y_j^i is associated with the object A at instant t and the parameters of the dynamic model are updated according to y_j^i.

Prediction is based on the use of a dynamic model which parameters are generally fixed by learning from a training sequence to represent plausible motions such as constant velocity or critically damped oscillations [4, 5]. For complex dynamics, such as non-constant velocities or non-periodic oscillations, the choice of the parameters for an estimation algorithm is difficult. Furthermore, the learning step becomes particularly more difficult in the case of missing data, where the dynamic between two successive observations is unknown. For these reasons, the parameters of our dynamic model are set in an adaptive and automated way once a measurement is available [1].
The energy $(E_k)_{y_j^i}$ contains three components, $\{E^1, E^2, E^3\}$, as defined below:

1. The Mahalanobis distance energy, $(E_k^1)_{y_j^i}$, measures the distance between a measurement y_j^i occurred at instant t and the prediction of the object A at $t-1$. This energy is sufficient to associate the available measurement if the object's motion is limited to translation in one direction (case of linear displacement). It is given by:

$$(E_k^1)_{y_j^i} = \sqrt{(y_j^i - \hat{A}(t-1))^T \hat{\Sigma}_k^{-1} (y_j^i - \hat{A}(t-1))}$$

   where Σ_k is the innovation covariance of the k^{th} object (we designe the k^{th} object by A in the equation).
2. To consider the case of complex dynamics, such as oscillatory motions or non-constant velocities, we have added the absolute accuracy evolution energy $(E_k^2)_{y_j^i}$. It introduces the notion of the geometric accuracy between two sets of features whose dynamic evolution is different. The description of both models are followed:
   – The updated dynamic model set considers that the available measurement y_j^i at t is generated by the k^{th} object and updates the parameters of its dynamic model to predict the new state of the object k at $t+1$;
   – The not updated dynamic model set predicts the new state at $t+1$ without considering the presence of the measurement, *i.e.* without updating the parameters of the dynamic model.

   $(E_k^2)_{y_j^i}$ extends a numerical estimation of the closeness between two dynamic model. Our idea aims to evaluate the parameters of the dynamic model in two cases if the measurement y_j^i arises from this object or no. We first predict the states $\hat{A}_1(t+1)$ and $\hat{A}_2(t+1)$ of the object at $t+1$. We then determine S_1, the intersection surface between the two circumscribed circles of the triangles $(\hat{A}(t-2), \hat{A}(t-1), \hat{A}(t))$ and $(\hat{A}(t-1), \hat{A}(t), \hat{A}_1(t+1))$, and S_2, the intersection surface between the two circumscribed circles of the triangles $(\hat{A}(t-2), \hat{A}(t-1), y_j^i)$ and $(\hat{A}(t-1), y_j^i, \hat{A}_2(t+1))$, (see Figures 1.(b-c)). $(E_k^2)_{y_j^i}$ is minimized when the similarity between both dynamic models is maximized and is given by:

$$(E_k^2)_{y_j^i} = |S_1 - S_2| \tag{2}$$

We compare these two sets to measure the ratio of similarity, defined by $R_s = 1 - (E_k^2)_{y_j^i}$, between the predictions at $t+1$ given by two different dynamic models for object k. Increasing this ratio maximizes the probability that the measurement y_j^i is generated by object k and the resemblance between two dynamic models.

A question might be asked: is the component E^2 able to handle all type of motions?

Indeed, E^2 evaluates a numerical measure of similarity between dynamic models. This measurement depends on the difference between two surfaces. It is considered as reliable if both positions, $\hat{A}(t)$ and y_j^i, are on the same side comparing to axis $(\hat{A}_{t-2}\hat{A}_{t-1})$, see Figure 1.(d). In Figure 1.(e), we show the case where both surfaces, S_1 and S_2, are quite similar, which imply E^2 to be null. This case can occur when the position of $\hat{A}(t)$ and y_j^i are diametrically opposite or when their positions are in different side comparing to axis $(\hat{A}_{t-2}\hat{A}_{t-1})$. In such cases, the energy E^2 is not a sufficient informative source to achieve the task of association. To compensate this energy, we incorporate the third energy E^3.

3. The proximity energy evolution, $(E_k^3)_{y_j^i}$, is the inverse of the surface S defined by the common area between the two triangles $(\hat{A}(t-2),\hat{A}(t-1),y_j^i)$ and $(\hat{A}(t-2),\hat{A}(t-1),\hat{A}(t))$ (see the dotted area of Figure 1.(f)). This energy evaluates the absolute accuracy between the prediction $\hat{A}(t)$ and the measurement y_j^i at instant t. Increasing S means that the prediction and measurement at instant t are close. This energy is given by:

$$(E_k^3)_{y_j^i} = \frac{1}{S} \tag{3}$$

Another question could be asked: why we use the intersection surface instead of only calculating the distance between the measurement y_j^i and the prediction of object's position at instant t?

In Figure 1.(g), we have two predictions at instant t, $\hat{A}_1$ and $\hat{A}_2$. They are both equidistant from the measurement y_j^i. If we only compute the distance to measure the proximity energy, we will get that both models have the same degree of similarity with the initiation model defined by the dynamic model of points $(\hat{A}(t-2),\hat{A}(t-1),y_j^i)$. This result leads to a contradiction with the reality. This problem can be explained by the fact that if they have both the same degree of similarity with the third dynamic model, we can conclude that their corresponding objects have the same dynamic. For this reason, we have chosen to evaluate the similarity by extracting the intersection surface between triangles. We can remark in Figure 1.(g) that these intersection surfaces are very different, which leads to a different measure in the degree of similarity.

Finally, the measurement y_j^i is associated with the object k if its energy magnitude is minimized:

$$\mathcal{Y}_{y_j^i \to k} = \left\{ \min_{k=1,\ldots,K} \left(\frac{1}{\sqrt{3}} \sqrt{\sum_{l=1}^{3} \alpha_l^2 (E_k^l)_{y_j^i}^2} \right) \right\}$$

with $0 \leq \alpha_l \leq 1$ and $0 \leq (E_k)_{y_j^i} \leq 1$.

We have described a novel approach for data association based on the minimization of an energy magnitude whose components are extracted from geometrical representations (area and distance) constructed with measurement, previous states and predictions. The purpose of choosing a geometrical definition for these energies refers to:

- show the geometrical continuity of the system between predictions and previous states using two different dynamic models;
- measure the similarity between predictions, at a particular time for the same object, using two different dynamic models, that logically must be quite similar because they represent the same system.

3 Experimental Results

3.1 Synthetic Test

To expose the performance of our energy minimization approach, we suggest the synthetic example of figure 2, which explores the case of oscillatory motion with a constant phase. It shows two objects T_1 and T_2 whose dynamic models are defined by two different sinusoids, $\sin(x)$ and $\sin(2x)+0.5$. The measurement y (full square in Figure 2), is equidistant from both objects and falls in their validation regions. In such case, both objects are candidates to be associated with this measurement. We compute the energy magnitude for each object (see Table 1) and obtain that $(E_1)_y < (E_2)_y$ and the measurement is associated with T_1.

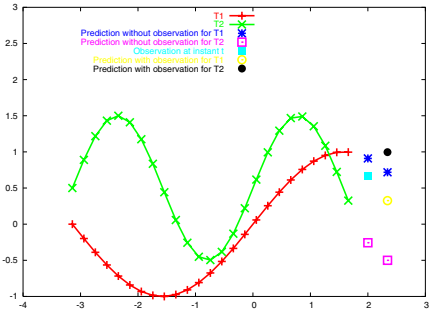


Fig. 2. Sinusoids with a constant phase: the dotted lines represent the trajectories of objects T_1 and T_2. The full square is the measurement y. The dotted squares and blue stars are the prediction of T_2 and T_1 at instants $\{t+1, t+2\}$ without taking into account the presence of the observation. The dotted and full circles are the prediction of T_1 and T_2 if we consider that the observation is associated with both objects.

We give another example where the motion of both objects is given by a sinusoid with a non periodic phase, see Figure 3. At instant $t-1$, both objects have the same position as shown in Figure 3. If we use the NNSF method, the measurement will be associated with both objects since it is equidistant from them. To improve the association result, we compute the energy magnitude for each object and the results are in Table 2. We obtain $(E_1)_y < (E_2)_y$ and the measurement is associated with T_1.

Table 1. Energy magnitude computed for objects T_1 and T_2

k	$\alpha_i(E_k^i)_y$	$(E_k)_y$
1	0.5 0.0001 0.4821	0.5278
2	0.5 0.9999 0.5179	0.9362

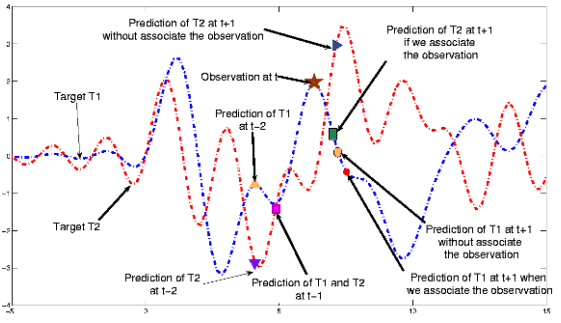


Fig. 3. Sinusoids with non-constant phase: the dotted lines represent the trajectories of objects; the shapes (square, circle, triangles) are their predictions at different instant

3.2 Van-Plane Test

In the following experiment, the available observation at instant t contains two measurements M_1 and M_2, each one represents a position in the object space (x,y). The first row in Figure 4 contains the frames at $\{t-2,t-1,t\}$ where two objects $\{T_1,T_2\}$, the van and the plane, are present. If we look at the position of these measurements on the real frame at t (right image in Figure 4), we observe that M_1 is closed to T_1 and M_2 to T_2. In the second and third rows, we show the prediction of both objects by evaluating two different dynamic model in the object space (x,y). We point out that the horizontal and the vertical axis of the frame are represented by the y-axis and x-axis in the object space. We can remark that the distance from T_1 to M_1 is larger than the one from T_1 to M_2, see the Mahalanobis distance energy $\alpha_1(E_1^1)_{\{M_1,M_2\}}$ in Table 3. Hence, if we use the Nearest Neighbor association method, the object T_1 will be associated to M_2 which causes a contradiction with the reality. To remedy, we compute the energies E^2 and E^3 which compensate E^1. Using the energy minimization approach, we obtain that the energies magnitude are minimized when M_1 and M_2 are respectively associated to objects T_1 and T_2 (see Table 3).

3.3 Walking Men Test

The "Walking men" sequence shows three close men walking at intant $t-2$, Figure 6.a. At $t-1$, two men (T_2 and T_3) continue in the same direction and the third one (T_1) takes the opposite direction, Figure 6.b.

The available observation at t contains three measurements given by a position in the object space (x,y). In Figure 6.c, we show the corresponding frame at instant t where we observe a partial occlusion between two walking men. We have put the measurements

Table 2. Energy magnitude computed for objects T_1 and T_2

k	$\alpha_i(E_k^i)_y$			$(E_k)_y$
1	0.5	0.0970	0.3094	0.3441
2	0.5	0.9030	0.6906	0.7170

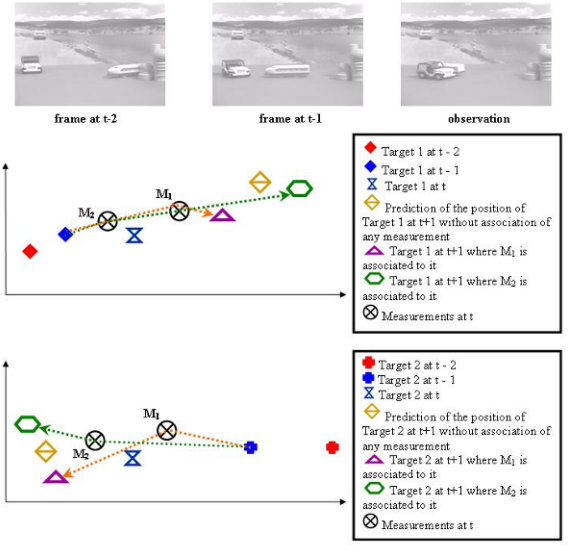


Fig. 4. Van Plane test: In the first row: left and middle image represent the frames at $\{t-2, t-1\}$ where two objects $\{T_1, T_2\}$ are present; right image is the real frame at t. The available observation at t contains two measurements $\{M_1, M_2\}$. In the second and third rows, we show the position of both objects at different instants.

Table 3. Energy magnitude computing for both objects when the measurements $\{M_1, M_2\}$ are associated with them

k	$\alpha_i(E_k^i)_{M_1}$			$(E_k)_{M_1}$
1	0.3141	0.02	0.3429	0.2687
2	0.6858	0.98	0.6571	0.7879

k	$\alpha_i(E_k^i)_{M_2}$			$(E_k)_{M_2}$
1	0.3179	0.7236	0.8636	0.6759
2	0.6821	0.2764	0.1364	0.4322

in Table 4 in a way that the measurement M_i is generated by the object T_i. We notice that the observed positions of the cross men are very near. Figure 6.d represents the real frame at instant $t+1$ where we remark that the cross men change their directions. We remark from Table 4 that the measurement M_2 is equidistant from objects T_2 and T_3 ($\alpha_2(E_2^2)_{M_2} = \alpha_2(E_3^2)_{M_2} = 0.15$). We also remark from Table 4 that most of time the

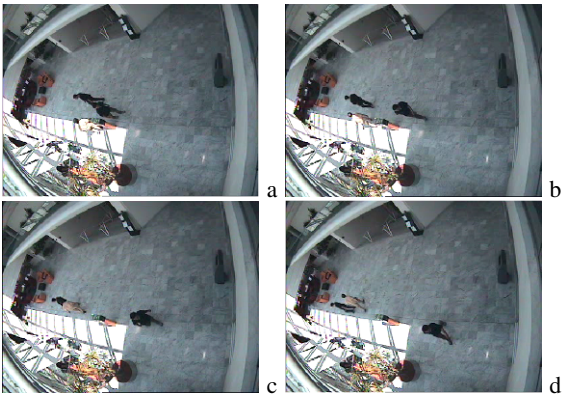


Fig. 5. Walking men test: (a) Frame at $t-2$ where we have three men close one to the other; (b) Frame at $t-1$ where two men continue in the same direction and the third takes the opposite direction; (c) The observation at t where two men cross and we have a partial occlusion between them; (d) Frame at $t+1$ where the cross men change their directions

Table 4. Energy magnitude computing for objects k when measurements M_i are associated with them

k	$\alpha_i(E_k^t)_{M_1}$			$(E_k)_{M_1}$
1	0.144	0.03	0	**0.085**
2	0.411	0.82	0	0.53
3	0.444	0.16	0	0.47
k	$\alpha_i(E_k^t)_{M_2}$			$(E_k)_{M_2}$
1	0.70	0.16	0.5	0.51
2	0.15	0.22	0.5	**0.32**
3	0.15	0.62	0	0.37
k	$\alpha_i(E_k^t)_{M_3}$			$(E_k)_{M_3}$
1	0.63	0.0037	0	0.363
2	0.21	0.76	0	0.455
3	0.16	0.235	0	**0.16**

third energy is null, this effect is due to the presence of a linear movement (motion limited to a displacement in two directions x and y). Once the energies are computed, we obtain the energy magnitude (E_i) is minimized when measurement M_i is associated to object T_i, see the column of $(E_k)_{M_i}$ in Table 4. Despite the change in illumination, the measurements were correctly associated to objects by using the approach of energy minimization.

3.4 Window Men Test

In this sequence, at instant $t-2$ we have two objects where the first does not move and the second undergoes a linear movement, Figure 6.a. At instant $t-1$, only the second

Fig. 6. Window men test: (a) Frame at $t-2$ where two objects are present; (b) Frame at $t-1$ where both objects are near; (c) the available observation at t; (d) Frame at intant t where both objects move

object continues to move and approaches from the first one, Figure 6.b. At instant t, an observation containing two measurements,defined by positions, is available. Figure 6.c represents its corresponding frame. Notice that the observed positions are near and there is a partial occlusion between both objects. In Table 5, we have the distance from measurement M_2 to object T_1 is less then the distance from M_2 to T_2, $(E_1^1)_{M_2} < (E_2^1)_{M_2}$, which leads to a contradiction with the reality. We compute the second energy to compensate the first one. The third energy is null due to the linear displacement that have both objects. As we can remark from Table 5, the energies magnitude $(E_k)_{M_1}$ and $(E_k)_{M_2}$ are minimized when M_1 and M_2 are associated to T_1 and T_2 respectively. We remark that our approach of energy minimization gives a correct association in spite of the presence of light reflexion on the window.

Table 5. Energy magnitude computing for objects k when measurements M_i are associated with them

k	$(E_k^i)_{M_1}$	$\alpha_i(E_k^i)_{M_1}$	$(E_k)_{M_1}$
1	0 0 0	0 0 0	0
2	0.0607 6.2 0	1 1 0	0.82
k	$(E_k^i)_{M_2}$	$\alpha_i(E_k^i)_{M_2}$	$(E_k)_{M_2}$
1	0.0217 0.36 0	0.3 0.83 0	0.51
2	0.0518 0.07 0	0.7 0.16 0	0.42

3.5 Ant Sequence Test

We have tested our method on another sequence where ant's motion is complex. They move with a non-constant velocity and can accelerate, decelerate and sometimes stop moving or starting. These ants are quite similar even non-distinguishable and characterized by the same gray level distribution. The sensor, at instant t, provides an observation containing six measurements corresponding to positions in the (x,y) space. In

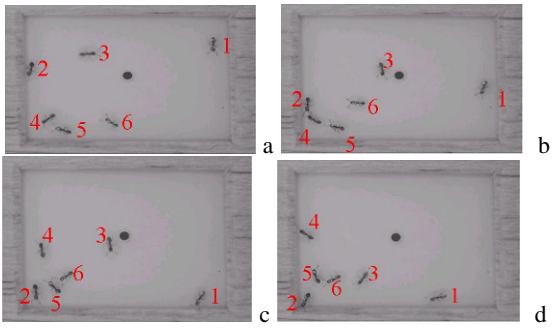


Fig. 7. Ants sequence: frames $\{10, 25, 35, 45\}$. (a-b) Acquisitions at $t-2$ and $t-1$; (c) Available observation at t; (d) Frame at $t+1$.

such scene, only one information could be used: the motion. We remark from Figure 7 that their displacement is erratic. The ants change their direction, accelerate, decelerate, stop moving, do rotation around their axis, *etc*. Figure 7.(a-b) are the acquisitions at instant $t-2$ and $t-1$ and represent the frames 10 and 25 of the ant sequence. We remark there is a considerable interval of time between these two frames. We have labeled the ants just to show their displacements from one frame to another. Figure 7.c shows the available observation at t and represents the frame 35 from the sequence. Figure 7.d is the real frame at $t+1$. Table 6 contains the numerical values of all energies between measurements and objects. We have multiplied each one by 100 to show clearly the difference between them. We have also fixed the order of classification in Table 6 so that the measurement M_i is provided from object T_i. If we use the Nearest Neighbor to associate the available observation, the measurements $\{M_2, M_4, M_5\}$ will be respectively associated to $\{T_4, T_2, T_2\}$ which leads to a contradiction with the reality (see the energy $\alpha_1(E_k^1)_{M_i}$ in Table 6). To remedy, we associate our observation by using the energy minimization approach. We remark from Table 6 that the energies $\alpha_2(E_2^2)_{M_2} < \alpha_2(E_4^2)_{M_2}$ and $\alpha_3(E_2^3)_{M_2} < \alpha_3(E_4^3)_{M_2}$ which compensate the error given by $\alpha_1(E_2^1)_{M_2}$. Finally, the following result is obtained: $(E_k)_{M_2}$ is minimized when M_2 is associated with object T_2. We recite that a measurement is associated with a object if the magnitude of its energy is minimal (equation 4). Lets take another example to show the necessity of using the energy E^3 in our formulation to compensate the others one. If we only use the energies $\alpha_1(E_k^1)_{M_i}$ and $\alpha_2(E_k^2)_{M_i}$ to associate data, we will get the following result: $(E_6)_{M_5} < (E_5)_{M_5}$ and the measurement M_5 will be associated with object T_6 which leads to an error in association. We can remark from Table 6 that $\alpha_3(E_5^3)_{M_5} < \alpha_3(E_6^3)_{M_5}$ which compensate the other energies. Finally, we observe that each measurement is well associated with its corresponding object. We notice that our approach is not a time-consumer. The total time of computation of all these energies is 0.25 seconds. We have used matlab to implement our method.

Table 6. First column and first row contain ant's numbers and measurement's numbers. The energies magnitude are multiplied by 100.

	$\alpha_1(E_k^1)_{M_i} \times 100$					
	1	2	3	4	5	6
1	6.512	47.239	25.761	43.975	48.705	46.618
2	22.545	5.762	21.381	**6.498**	**2.510**	12.014
3	15.105	24.891	**4.043**	17.747	23.748	19.693
4	21.447	**1.728**	21.604	9.403	5.444	10.209
5	18.317	6.094	17.943	12.191	**9.276**	5.953
6	16.074	13.549	9.268	10.923	**10.318**	5.513

	$\alpha_2(E_k^2)_{M_i} \times 100$					
	1	2	3	4	5	6
1	1.487	1.848	1.424	1.081	11.276	1.091
2	3.096	1.234	3.091	9.610	54.460	1.583
3	14.168	0.234	0.343	0.160	1.256	0.107
4	6.564	2.211	2.669	0.307	24.120	4.400
5	74.366	85.842	92.240	96.185	**6.993**	92.438
6	0.320	0.255	0.233	1.032	**1.895**	0.381

	$\alpha_3(E_k^3)_{M_i} \times 100$					
	1	2	3	4	5	6
1	0.037	45.813	0.792	14.091	48.144	43.240
2	6.801	0.014	24.593	0.225	14.570	13.635
3	22.205	8.488	1.783	5.637	4.657	24.499
4	38.757	14.837	27.829	0.708	17.822	12.073
5	8.457	10.537	12.624	3.231	**4.048**	6.203
6	23.743	20.101	32.380	76.319	**10.759**	0.350

	$(E_k)_{M_i} \times 100$					
	1	2	3	4	5	6
1	**3.856**	38.008	14.903	26.668	40.071	36.715
2	13.713	**3.402**	18.899	6.699	32.580	10.532
3	17.530	15.184	**2.559**	10.751	13.991	18.148
4	25.853	8.718	20.399	**5.447**	17.598	9.475
5	44.487	50.057	54.741	56.008	**7.103**	53.600
6	16.555	13.996	19.446	44.516	8.676	**3.197**

4 Conclusions

This work proposes a new method for data association based on an energy minimization. The developed approach can handle complex motions and highly non-linear systems, and deals with the lack of of prior knowledge. Its effectiveness returns to the fact it requires few parameters. The geometric illustration of energy components allows to measure the accuracy between two dynamic models and to define their degree of

similarity. As a perspective for this work, we suggest to integrate the energy minimization approach within the classical particle filter to build a new framework for multiple tracking objects. Moreover, since we consider erratic motions that cannot be learned from training sequences, we suggest to use an adaptive and automated way to set the parameters of the dynamic model of the filter. The purpose of developing this framework is to track objects under the restriction of the missing of prior information and especially when similar objects are evolving in the scene. This phase currently is under development.

References

1. El Abed, A., Dubuisson, S., Béréziat, D.: Comparison of statistical and shape-based approaches for non-rigid motion tracking with missing data using a particle filter. In: Advanced Concepts for Intelligent Vision Systems (2006)
2. Vermaak, J., Godsill, S.J., Pérez, P.: Monte Carlo Filtering for Multi-Target Tracking and Data Association. IEEE Transactions on Aerospace and Electronic Systems (2005)
3. Rasmussen, C., Hager, G.D.: Probabilistic data association methods for tracking complex visual objects. IEEE Transactions on Pattern Analysis and Machine Intelligence (2001)
4. North, B., Blake, A., Isard, M., Rittscher, J.: Learning and classification of complex dynamics. IEEE Transactions on Pattern Analysis and Machine Intelligence (2000)
5. Blake, A., Isard, M.: Active Contours. Springer, Heidelberg (1998)
6. Rong, L.X., Bar-Shalom, Y.: Tracking in clutter with nearest neighbor filter: analysis and performance. IEEE transactions on aerospace and electronic systems (1996)
7. Rago, C., Willett, P., Streit, R.: A comparison of the JPDAF and MHT tracking algorithms. IEEE Acoustics, Speech and Signal Processing (1995)
8. Fortmann, T., Bar-Shalom, Y., Scheffe, M.: Sonar tracking of multiple targets using joint probabilistic data association. IEEE Journal Oceanic Engineering (1983)

Author Index