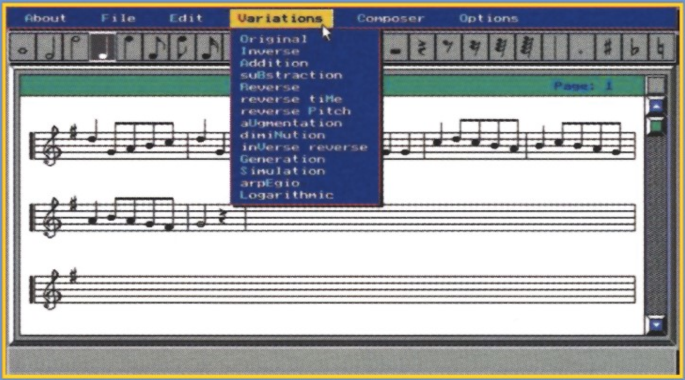
Hot Topics

LNCS 3310

Uffe Kock Wiil (Ed.)

Computer Music Modeling and Retrieval

Second International Symposium, CMMR 2004
Esbjerg, Denmark, May 2004
Revised Papers



Springer

EXTRA MATERIALS
extras.springer.com

Uffe Kock Wiil (Ed.)

Computer Music Modeling and Retrieval

Second International Symposium, CMMR 2004
Esbjerg, Denmark, May 26-29, 2004
Revised Papers

Springer

Volume Editor

Uffe Kock Wiil
Aalborg University Esbjerg
Department of Software and Media Technology
Niels Bohrs Vej 8, 6700 Esbjerg, Denmark
E-mail: ukwiil@cs.aue.auc.dk

Library of Congress Control Number: 2004118126

CR Subject Classification (1998): H.3, H.4, H.5, H.2, I.2, C.3

ISSN 0302-9743
ISBN 3-540-24458-1 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Boller Mediendesign
Printed on acid-free paper SPIN: 11381006 06/3142 5 4 3 2 1 0

Preface

This volume contains the final proceedings for the 2004 Computer Music Modeling and Retrieval Symposium (CMMR 2004). This event was held during 26–29 May 2004 in Esbjerg, Denmark on the joint campus area of Aalborg University Esbjerg and the University of Southern Denmark, Esbjerg. CMMR is an annual event focusing on important aspects of computer music. CMMR 2004 is the second event in this series. CMMR 2003, which was held in Montpellier, France in May 2003, was a great success and attracted high-quality papers and prominent researchers from the field of computer music. The CMMR 2003 postsymposium proceedings was published by Springer in the Lecture Notes in Computer Science series (LNCS 2771). CMMR 2004 was jointly organized by Aalborg University Esbjerg in Denmark and LMA, CNRS, Marseille in France (in cooperation with ACM SIGWEB).

The use of computers in music is well established. CMMR 2004 provided a unique opportunity to meet and interact with peers concerned with the cross-influence of the technological and creative in computer music. The field of computer music is interdisciplinary by nature and closely related to a number of computer science and engineering areas such as information retrieval, programming, human computer interaction, digital libraries, hypermedia, artificial intelligence, acoustics, signal processing, etc. The event gathered many interesting people (researchers, educators, composers, performers, and others). There were many high-quality keynote and paper presentations, that fostered inspiring discussions. I hope that you find the work presented in these proceedings as interesting and exciting as I have.

First of all, I would like to thank the Program Chair Richard Kronland-Martinet for the very fruitful cooperation during the organization of this second event in the CMMR series. I would also like to thank my colleagues Laura Hyland, Stefania Serafin, and Lars Graugaard for their help in organizing the event. Finally, this volume would not have been possible without the help of Springer, Heidelberg. In particular, I would like to thank the computer science editor, Christine Günther, and the executive editor of the LNCS series, Alfred Hofmann.

October 2004 Uffe Kock Wiil

Organization

CMMR 2004 was jointly organized by Aalborg University Esbjerg, Denmark and LMA, CNRS, Marseille, France.

Symposium Chair

Uffe Kock Wiil (Aalborg University Esbjerg, Denmark)

Local Arrangements and Publicity

Uffe Kock Wiil (Aalborg University Esbjerg, Denmark)
Laura Hyland (Aalborg University Esbjerg, Denmark)
Stefania Serafin (Aalborg University Esbjerg, Denmark)
Lars Graugaard (Aalborg University Esbjerg, Denmark)

Program Committee

Chair:	Richard Kronland-Martinet (LMA, CNRS, Marseille, France)
Members:	Jens Arnspang (Aalborg University Esbjerg, Denmark)
	Philippe Depalle (McGill University, Montreal, Canada)
	Barry Eaglestone (University of Sheffield, UK)
	Anders Friberg (Royal Institute of Technology, Sweden)
	Goffredo Haus (University of Milan, Italy)
	David L. Hicks (Aalborg University Esbjerg, Denmark)
	Henkjan Honing (University of Amsterdam, The Netherlands)
	Kristoffer Jensen (University of Copenhagen, Denmark)
	Matti Karjalainen (Helsinki University of Technology, Finland)
	Henrik Legind Larsen (Aalborg University Esbjerg, Denmark)
	Brian Mayoh (University of Aarhus, Denmark)
	Jocelyne Nanard (LIRMM, Montpellier, France)
	Marc Nanard (LIRMM, Montpellier, France)
	Peter J. Nürnberg (Aalborg University Esbjerg, Denmark)
	François Pachet (Sony Research Lab, France)
	Violaine Prince (LIRMM, Montpellier, France)
	Esben Skovenborg (TC Electronic, Denmark)
	Julius Orion Smith III (Stanford University, USA)
	Leonello Tarabella (CNR, Pisa, Italy)
	Daniel Teruggi (INA, Paris, France)
	Hugues Vinet (IRCAM, Paris, France)
	Gerhard Widmer (University of Vienna, Austria)
	Sølvi Ystad (LMA, CNRS, Marseille, France)

Music Selection

Chair:	Lars Graugaard (Aalborg University Esbjerg, Denmark)

Sponsoring Institutions

Aalborg University Esbjerg, Denmark
University of Southern Denmark, Esbjerg, Denmark
Vestjysk Musikkonservatorium (Esbjerg Music Conservatory), Denmark
Seminariets Uddannelsesfond, Esbjerg, Denmark
LMA, CNRS, Marseille, France

Table of Contents

Separating Voices in Polyphonic Music: A Contig Mapping Approach

Elaine Chew and Xiaodan Wu

University of Southern California,
Viterbi School of Engineering, Integrated Media Systems Center,
Epstein Department of Industrial and Systems Engineering,
3715 McClintock Avenue GER240 MC:0193,
Los Angeles, California, USA
{echew, xiaodanw}@usc.edu

Abstract. Voice separation is a critical component of music information retrieval, music analysis and automated transcription systems. We present a contig mapping approach to voice separation based on perceptual principles. The algorithm runs in $O(n^2)$ time, uses only pitch height and event boundaries, and requires no user-defined parameters. The method segments a piece into contigs according to voice count, then reconnects fragments in adjacent contigs using a shortest distance strategy. The order of connection is by distance from maximal voice contigs, where the voice ordering is known. This contig-mapping algorithm has been implemented in VoSA, a Java-based voice separation analyzer software. The algorithm performed well when applied to J. S. Bach's *Two- and Three-Part Inventions* and the forty-eight Fugues from the *Well-Tempered Clavier*. We report an overall average fragment consistency of 99.75%, correct fragment connection rate of 94.50% and average voice consistency of 88.98%, metrics which we propose to measure voice separation performance.

1 Introduction

This paper presents an algorithm that separates voices in polyphonic music using basic principles of music perception and proposes metrics for evaluating the correctness of the machine-generated solutions. Creating music with multiple voices that are relatively independent is a compositional technique that results in auditory pleasure and has been practised for centuries in western music. This has led to a library of compositional rules that facilitate auditory streaming and the perception of multiple voices dating as far back as Palestrina (1526-1594) and as recently as Huron (2001, see [7]). In this paper, we use knowledge of the perceptual principles of auditory streaming to create an $O(n^2)$ contig mapping algorithm for separating polyphonic pieces into their component voices.

Distinct from audio source separation, voice separation is the determining of perceptible parts or voices from multiple concurrently sounding streams of music. The multiple streams can originate from the same source and also be of the same timbre. The contig mapping approach described in this paper considers only pitch height and event boundaries, ignoring information on timbre and sound source. Prior researchers (such as [8], [11] and [2]) have not reported any significant testing on large corpora because of the lack of methods for quantitative evaluation of voice separation results. We propose three metrics for quantifying the goodness of voice separation results and test the contig mapping algorithm on Johann Sebastian Bach's 15 *Two-Part Inventions*, 15 *Three-Part Inventions* and 48 Fugues from the *Well-Tempered Clavier*.

Computationally viable and robust methods for voice separation are critical to machine processing of music. Separating music into its component voices is necessary for notating music in separate staffs according to voice or instrument, or in the same staff with stems up or down depending on voice [8]. Another application related to music transcription is pitch spelling, the assignment of letter names to numeric representations for pitches or pitch classes (see for example, [3], [4] and [10]). The spelling of any given pitch is based on its tonal context as well as accepted voice leading principles. Voice separation is a precursor to incorporating voice leading spelling rules to any pitch spelling method.

Many applications in music information retrieval require the matching of *monophonic* queries to *polyphonic*[1] (or *homophonic*) databases, for example, query by humming applications. While other approaches to matching single line queries to multi-line records exist (see for example [9]), one approach made possible by voice separation is to first separate each piece into its component voices prior to matching the melodic query to now single-line records. Hence, a robust voice separation algorithm will vastly improve the hit rate of matching melodic queries to polyphonic databases. Another computational problem relevant to music information retrieval is the automatic detection and categorization of music by meter. Metric structure is most obvious in the lower voices and methods for meter induction can be improved by voice separation techniques.

The final example of a voice separation application is that of expressive performance. One of the main tasks of the performer or conductor is to determine the main melody or motif in any musical segment. The notes in the segment to be highlighted is often played louder or even a little before the others that are notated simultaneously in the score [6]. At other times, different voices are sounded at different volume levels to produce a foreground and background effect. Hence, machine models for voice separation are also essential to knowledge-based approaches to generating expressive performances.

As shown above, voice separation is a valuable tool in music information retrieval, automated transcription and computer analysis of music. One of the

[1] In traditional music literature, there exists a clear distinction between *polyphony* and *homophony*. Polyphonic music is multi-voice music where the voices exhibit independence relative to one another. Homophonic music, although also consisting of multiple voices, has one primary lead voice while other voices act as accompaniment to the main melody. In contrast, *heterophonic* music (less well defined) is music with one primary melody, and all accompanying voices embelishing with variants of the main theme.

easiest approaches to voice separation is to split voices according to some set of non-overlapping pitch ranges. According to [8], this is the method adopted by most commercial sequencer software packages. Needless to say, this method of separating voices can produce highly inaccurate and unsightly (in the case of automatic transcription) results. Various researchers have proposed ways to improve on this primitive approach.

In [11], Temperley proposed a preference rule approach to voice separation, incorporating the following rules for assigning voices to piano-roll representation of music: 1. avoid large leaps in any one stream; 2. minimize the number of streams; 3. minimize long breaks in streams; 4. avoid having more than one stream occupy a single square; and, 5. maintain a single top voice. Rules 1 through 4 were tested on four of Bach's fugues. Rule 5 was found to be necessary for handling classical music; rules 1 through 5 were tested on a few classical string quartets. The errors were analyzed in terms of the number of breaks, missed or incorrect collisions and misleads. Another rule-based approach was briefly described by Cambouropoulos in [2]. This method segments the input into beats then, within each beat, connects all onsets into streams by selecting the shortest path. The crossing of streams is disallowed and the number of streams is set to be equal to the number of notes in the largest chord.

In [8], Kilian and Hoos proposed a local optimization approach to voice separation. The piece was first partitioned into slices which can contain parts that overlap (in time) with other slices. Within each slice, the notes are then separated into voices by minimizing a cost function, which assigns penalty values for undesirable features such as, overlapping notes and large pitch intervals. One flexible feature of the Kilian and Hoos model is the ability to assign entire chords to one single voice. (The cost function penalizes chord tones that are spread too far apart.) The penalty values can be adjusted by the user to achieve different tradeoffs between the features. Their algorithm was tested on selected Bach *Chorales* and Chopin *Valses*, and Bartok's *Mikrokosmos*, and was found to be sensitive to the choice of penalty function parameters. For the purpose of automated transcription, the user can change the parameter values until a satisfactory result is achieved.

Like Temperley, our goal is to produce a correct analysis rather than an appropriate one for transcription, as is the case for Kilian and Hoos. In this paper, we propose three metrics to measure the correctness of a voice separation solution. They are: the average fragment consistency, the correct fragment connection rate and the average voice consistency. These metrics allow the algorithm's results to be quantified objectively. Unlike Kilian and Hoos' local optimization approach, our method does not allow synchronous notes to be part of the same voice. On the other hand, the contig mapping approach exhibits high fragment consistency, the grouping of notes from the same voice into the same fragments.

Both Temperley's preference rule approach as well as Kilian and Hoos' local optimization approach can protentially incur prohibitive computational costs if all possible solutions were enumerated and evaluated. Temperley utilized dynamic programming while Kilian and Hoos used a heuristically-guided stochastic local search procedure to avoid the exponential computational cost of exhaustive enumeration. In contrast, the contig mapping approach has an $O(n^2)$ performance and does not require approximation methods to compute a solution.

Distinct from previous approaches, our method hinges on one important feature of polyphonic music that has been ignored by other researchers. Because voices tend not to cross, when all voices are present, one can be certain of the voice ordering and assignment. We use these maximal voice segments as pillars of certainty out of which each voice connects to other members of its stream. This method requires no pre-assigned parameters or rule definitions. The perceptual rules are incorporated into the mathematical model and the algorithm has a guaranteed worst case performance of $O(n^2)$.

Section 2 describes the perceptual principles and the concepts underlying the contig mapping approach, and introduces the contig mapping algorithm. Section 3 presents additional details of the computer implementation of the algorithm and describes the VoSA (Voice Separation Analyzer) software. Section 4 presents our evaluation techniques and computational results. Finally, Section 5 outlines our conclusions and future work.

2 The Contig Mapping Approach

This section presents the contig mapping approach and its underlying perceptual principles. Section 2.1 outlines the auditory perceptual principles relevant to our approach, and Section 2.2 extracts from the principles and rules the assumptions underlying the contig mapping algorithm. Section 2.3 describes the contig mapping algorithm, including the segmentation procedure and the fragment connection policy.

2.1 Perceptual Principles for Voice Leading

In this section, we highlight the perceptual principles that are relevant to the contig mapping approach. Because the goal of the rules of voice leading is to create two or more concurrent yet distinct parts or voices, the same rules result in optimal auditory streaming. In [7], Huron reviews the perceptual principles for the organizing of auditory stimuli into streams and derives the rules of voice leading from these principles and empirical evidence.

The first is the pitch proximity principle. In the review, Huron reports that Bregman and his colleagues have gathered strong evidence for the pre-eminence of pitch proximity over trajectory in stream organization [1]. He argues that "the coherence of an auditory stream is maintained by close pitch proximity in successive tones within the stream," and that this principle holds true in the music across different cultures. Thus, in determining the connections between notes that are perceived to be from the same stream, proximity should be the guiding principle.

The second is the stream crossing principle. Humans have great difficulty in tracking streams of sounds that cross with respect to pitch. Huron reports the results of Deutsch [5] who showed that concurrent ascending and descending streams of the same timbre are perceived to switch directions at the point of crossing[2] as shown in the diagram on the right in Figure 1. Hence, a guiding principle in connecting notes in the same stream is that the streams should not cross.

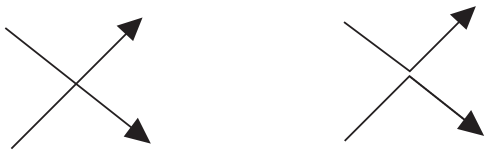


Fig. 1. Possible interpretations of crossing streams.

These perceptual principles lead to numerous traditional and non-traditional rules for writing polyphonic music with perceptibly distinct parts. The ones relevant related to the pitch proximity principle are (following Huron's numbering system):

[D6.] Avoid Unisons Rule. *Avoid shared pitches between voices.*
D10. Common Tone Rule. *Pitch-classes common to successive sonorities are best retained as a single pitch that remains in the same voice.*
D11. Conjunct Movement Rule. *If a voice cannot retain the same pitch, it should preferably move by step.*
C3. Avoid Leaps Rule. *Avoid wide pitch leaps.*
D13. Nearest Chordal Tone Rule. *Parts should connect to the nearest chordal tone in the next sonority.*
[D18.] Oblique Approach to Fused Intervals Rule. *When approaching unisons, octaves, or fifths, it is best to retain the same pitch in one of the voices.*
[D19.] Avoid Disjunct Approach to Fused Intervals Rule. *If it is not possible to approach unisons, octaves and fifths by retaining the same pitch, step motion should be used.*

while D6, D14 and D15 are encapsulated in the stream crossing principle:

[D6.] Avoid Unisons Rule. *Avoid shared pitches between voices.*
D14. Part-Crossing Rule. *Avoid the crossing of parts with respect to pitch.*

[2] A simple and informal experiment conducted on March 4th in a class of 14 students showed that this result held true even when the ascending and descending streams were played using the rhythm of the Christmas carol "Joy to the World," where the opening melody is essentially a descending scale embellished with temporal variation. This perceptual principle is so strong that it overrode the perception of the well-known melody.

D15. Pitch Overlapping Rule. *Avoid "overlapped" parts in which a pitch in an ostensibly lower voice is higher than the subsequent pitch in an ostensibly higher voice.*

2.2 The Assumptions and Underlying Concept

For the purpose of the contig mapping algorithm, we translate the rules and perceptual principles detailed in Section 2.1 to the following assumptions:

1. By definition, each voice can only sound at most one note at any given time.
2. All the voices will sound synchronously at some time (we use this as a baseline count of the total number of voices present in the piece.)
3. *Pitch Proximity:* intervals are minimized between successive notes in the same stream or voice.
4. *Stream Crossing:* voices tend not to cross.

The contig mapping approach derives its method directly from these assumptions. Assumptions 1, 2 and 4 imply that, at certain segments of time, all voices will sound synchronously in a well-bahaved manner. In these segments, which we call *maximal voice contigs*, we can be certain of the voice assignments for each note. Based on assumptions 3 and 4, we can use distance minimizing procedures to connect voices between segments. The maximal voice contigs seed the connection process: they act as the pillars out of which voice assignments grow at each iteration of our procedure.

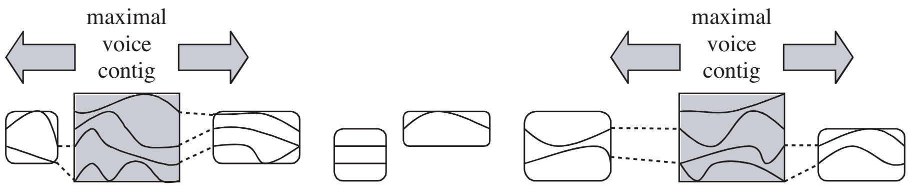


Fig. 2. Minimum distance voice connections grow out from the maximal voice contigs

2.3 The Algorithm

We have outlined the principles and concept behind our contig mapping approach in the previous sections. In this section, we shall provide the algorithmic details for its systematic implementation, including the procedures for segmentation and connection.

Before embarking on a description of the algorithm, we first introduce the terminology used in this section. A *note* is a musical entity with pitch and duration properties. A *fragment* is a sequence of successive notes that belong to the same voice. A *contig*[3] is a collection of overlapping fragments such that the overlap depth (number of fragments present) at any time is constant. A *maximal voice contig* is a contig with the maximum number of voices present. Examples of a fragment, contig and maximal voice contig are shown in Figure 4, which corresponds to bars 24 and 25 of Bach's Three-Part Invention (Sinfonia) No. 13 (shown in Figure 3.) In this case, both the first and last contigs are maximal voice contigs.

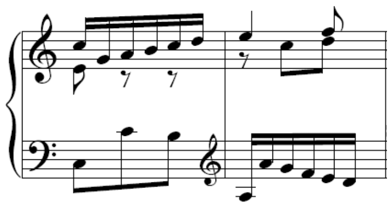


Fig. 3. Measures 24 and 25 of Bach's Three-Part Invention No.13.

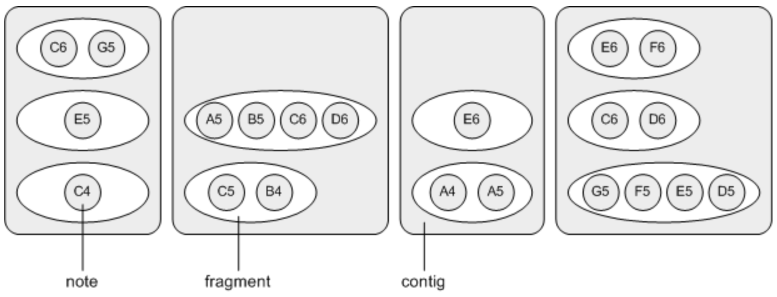
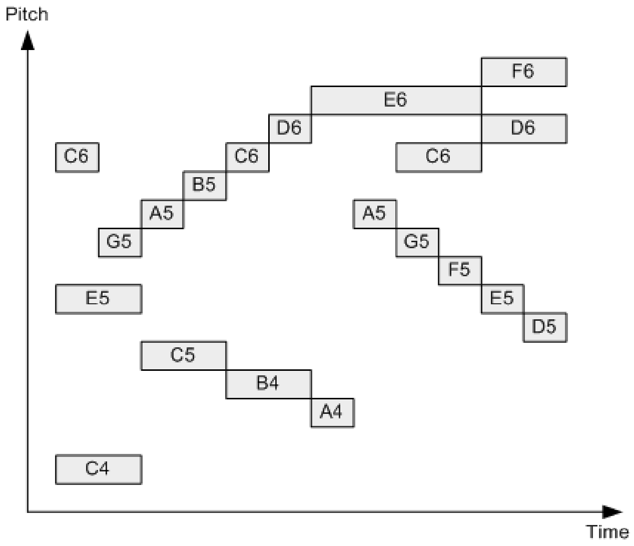


Fig. 4. Terminology

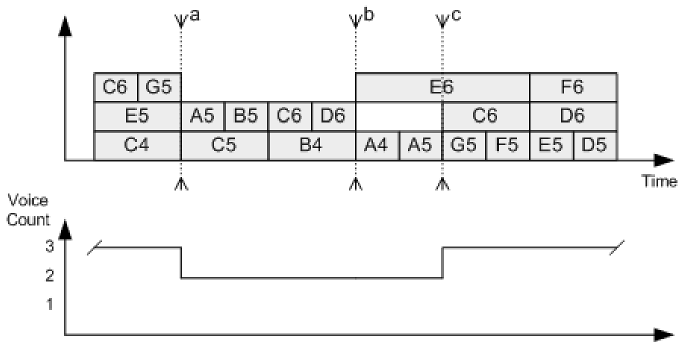
Segmentation Procedure The piece is segmented according to voice count. The segmentation procedure is best illustrated by example. The final outcome is a segmentation of the original piece into contigs such that the voice count remains constant within the contig. We return to the Bach Three-Part Invention example shown in Figure 3. Figure 5(a) shows a piano roll representation of the same

[3] The term *contig* is borrowed from the field of computational biology where, in DNA sequencing, the shotgun sequencing method utilizes computer algorithms to connect ordered sets of overlapping clones of DNA fragments in order to determine the DNA sequence.

excerpt. The lower half of Figure 5(b) charts the voice count at any given time while the upper half of the figure shows the flattened piano roll representation and the segmentation boundaries, marked as "a", "b" and "c." Boundaries a and c result from the change in voice counts, while boundary b is the results of the voice status change.



(a) piano roll representation



(b) flattened piano roll representation with segmentation, and voice count plot

Fig. 5. Example: Bach's Three-Part Invention No.13, measures 24 and 25.

More formally, if v_t represents the voice count at time slice t, the boundary between time slices $t-1$ and t becomes a segmentation boundary if:

- either $v_t \neq v_{t-1}$;
- or $v_t = v_{t-1}$ but the voice status changes.

A voice status change is caused by held notes that cross over a segmentation boundary, and thus are suspended over an empty slot as shown in the segment (b,c) in Figure 5(b). The held note resulted in a status change across boundary b even though the voice count does not change. As a result, b becomes a segmentation boundary. Because the note E6 crosses the boundary c, this note will be cloned, marked as being a part of a longer note and duplicated in the contigs on either side of boundary c. The resulting segmentation is shown in the contig diagram in Figure 4.

Connection Policy After segmentation, the maximal voice contigs seed the connection process. They act as the centers out of which the connections to neighboring contigs grow. Because voices tend not to cross and maximal voice contigs contain all voices, the voice assignment for each note in a maximal voice contig is known with high certainty. Hence, all fragments in each maximal voice contig are ordered by pitch height and assigned voice numbers corresponding to their ordering. In connecting voice fragments across neighboring contigs, we select the distance minimizing choice. Connected fragments are assigned to the same voice, and the fragment assembly process grows out from the maximal voice contigs.

Because the number of voices is usually small[4], we can enumerate all possible connection combinations and select the one with the lowest penalty. Suppose we wish to connect the fragments in two neighboring contigs, X and Y, where X is followed by Y (in time). Consider a note, q_X, that is the last one from a fragment in contig X and another, p_Y, that is a first note in a fragment in contig Y. The cost of connecting q_X to p_Y, $c(q_X, p_Y)$, is assigned based on the following rules:

- if the two notes are segments of the same longer note, $c(q_X, p_Y) = -2^{31}$;
- if one of the two notes is null or both, $c(q_X, p_Y) = 2^{31}$;
- else, $c(q_X, p_Y)$ is the absolute difference between the pitches of the two notes.

The first rule ensures that all long notes that were previously partitioned are re-connected at this stage. The second rule forces all connectible fragments to be assigned a partner whenever one exists. And the third rule ensures minimal distance assignments.

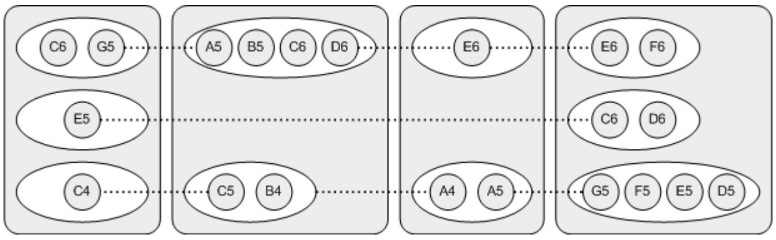
The connection sequence grows outward from the maximal voice contigs, which act as seeds for the connection procedure. First, fragments in the immediate neighbors are connected to those in each maximal voice contig (this first level connection is illustrated in Figure 2.) Then, the second order neighbors are connected to the immediate neighbors, and so on. The assembling procedure can be viewed as a crystallization process. The maximal voice contigs act as seeds

[4] According to Huron's Principle of Limited Density [7], "If a composer intends to write music in which independent parts are easily distinguished, then the number of concurrent voices or parts ought to be kept to three or fewer." Typically, the number of voices range from two to four, and occasionally, five or six voices are utilized. However, in the latter cases, the human ear cannot distinguish more than three or four concurrent voices at any given time.

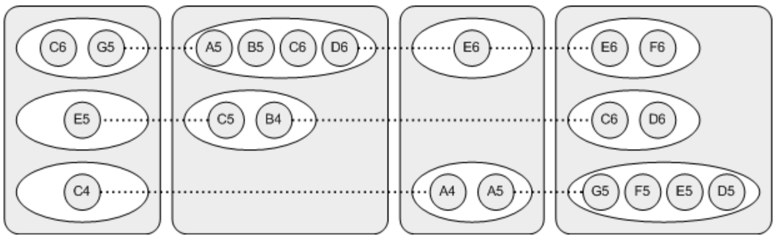
for the process, and the contigs closer to these seeds will be connected first. The procedure ends when all contigs (or fragments in every contig) are connected.

In a piece with n notes, there can be at most n contigs. At each iteration, at least one (and at most n) neighboring contig(s) is connected to a growing section centered around a maximal voice contig. There are at most n such iterations, hence the worst case complexity is $O(n^2)$.

The shortest distance connection policy produces correct groupings in the vast majority of cases. However, it is useful to note that sometimes the policy may not generate the correct solution. See, for example, the connection solutions presented in Figure 6. In the figure, dotted lines link fragments that are grouped into the same voice. The correct solution is shown in Figure 6(a) while the shortest distance solution is given in Figure 6(b). The algorithm assigns the lower fragment in the second contig to the incorrect voice. These erroneous connections are visually presented in Figure 8(b) as the four "X"'s on the left hand side. Because of the robustness of the maximal contig approach, this one incorrect assignment will not affect the majority of the notes, which are correctly grouped together according to voice.



(a) correct connections



(b) shortest distance connections

Fig. 6. Connection solutions for Bach's Three-Part Invention No.13, measures 24 and 25.

3 Implementation

The contig mapping approach to voice separation has been implemented in a Java application called VoSA, the Voice Separation Analyzer. The platform-independent application was developed under Java jdk1.4.2 and runs on Windows, Mac OS and Unix machines. Its graphical user interface allows the user to visualize and evaluate the results of the voice separation algorithm. The current version of VoSA takes only MIDI input. It also has the capacity to export voice separated pieces in MIDI format and evaluation results in comma separated value (CSV) format. In this section, we present the implementation strategies not covered in the previous section's explanation of the algorithm, and describe VoSA's graphical user interface.

3.1 Quantization

Because performance artifacts and rounding errors produce overlapping notes from the same voice or gaps between successive notes, we use a selective snapping procedure to quantize the data. Since we are not concerned with beat onset irregularities, quantization only needs to occur at the boundaries with ambiguous note overlaps or gaps between note boundaries. Unlike the usual quantizing procedure of snapping the observed note boundaries to the closest unit grid, the selective snapping will only be invoked when the time difference between any two note boundaries is less than a given threshold (we used 30ms). Figure 7 shows the selective snapping quantization procedure. After quantization, the notes of the piece are stored as an ordered list sorted by onset times.

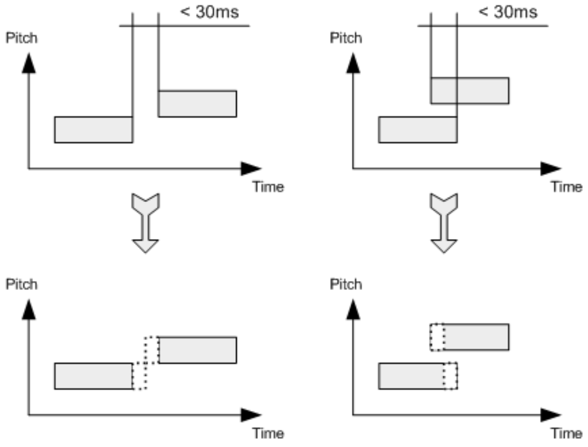


Fig. 7. The selective snapping quantization procedure.

3.2 Treatment of Ending Chordal Emblishments

In the library of contrapuntal pieces we tested, many of the polyphonic compositions have endings that are embellished with chords consisting of more notes than the number of voices in the pieces. These ending chords serve as statements of finality but also masquerade as maximal voice contigs, causing VoSA to overestimate the number of voices in the piece and also to grow the one maximal voice contig from right to left, a highly suboptimal process. To facilitate the search for the "true" maximal voice contigs, we exclude the last three contigs to compute the maximum number of voices, and eliminate all voice fragments with an index greater than the maximum voice count. These discarded fragments (a small fraction of the total notes in the piece) will not be counted during the evaluation process.

3.3 User Interface

VoSA provides a graphical user interface for the user to analyze the performance of the voice separation algorithm. This graphical user interface is shown in Figure 8. The upper part of the Figure 8(a) shows the piano roll representation and the segmentation of Bach's Three-Part Invention No.13. In the lower part of Figure 8, a graph charts the voice count at each point in time. The vertical lines in the piano roll graph shows the segmentation boundaries indexed by the contig numbers.

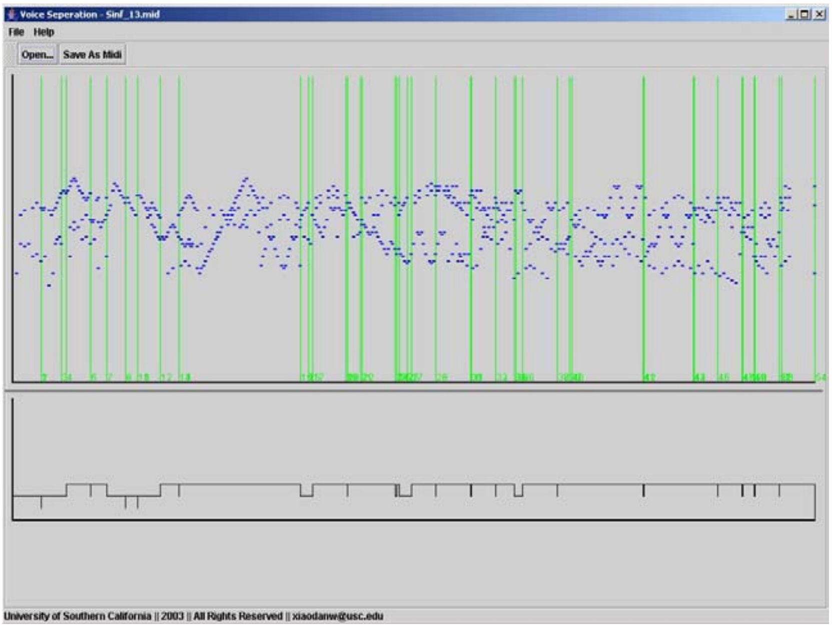
The latest version of VoSA, VoSA 3, incorporates zoom-in and zoom-out capabilities, colors voice assignments by voice, and marks the erroneous connections by a red "X." Figure 8(b) shows a screenshot of a zoomed-in analysis of the results of voice separation for Bach's Three-Part Invention No.13. The red X's mark the points at which connections were incorrectly assigned.

4 Computational Results

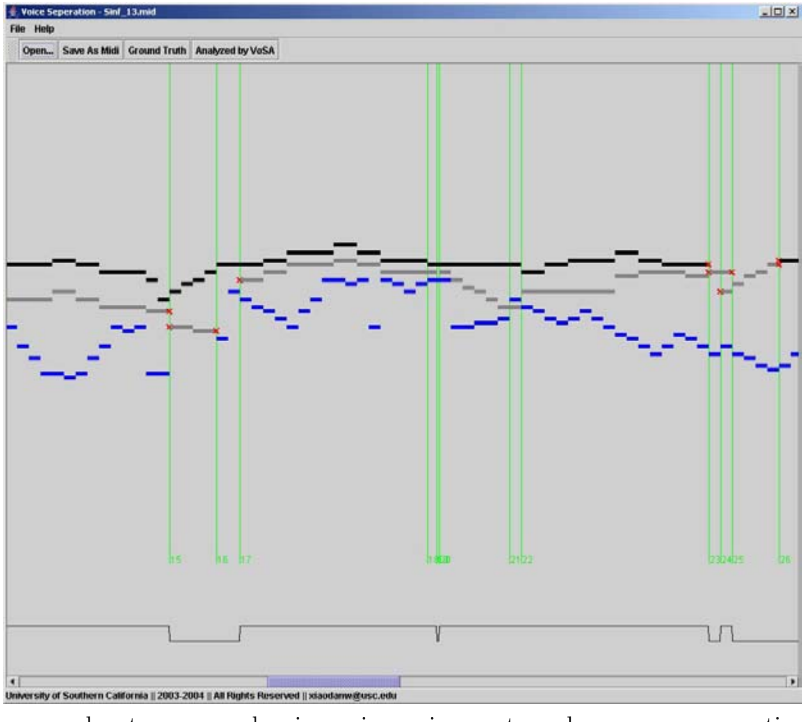
This section presents the contig mapping algorithm's voice separation results when applied to polyphonic music by J. S. Bach, namely his *Two- and Three-Part Inventions* and Fugues from the *Well-Tempered Clavier*. Section 4.1 describes the test corpus and the acquisition of voice separation solutions. Section 4.2 lays out the evaluation procedures and Section 4.3 presents the evaluation statistics for our test corpus.

4.1 Test Data and Ground Truth

We test the contig mapping algorithm using Johann Sebastian Bach's (1685-1750) 48 Fugues from his *Well-Tempered Clavier* (BWV 846-893), his *Two-Part Inventions* (BWV 772-786) and his *Three-Part Inventions* (BWV 787-801), also known as *Sinfonias*. As noted by Temperley in [11], "the correct 'contrapuntal analysis' for a piece is often not entirely clear. ... One case where the correct contrapuntal analysis is explicit is Bach fugues (and similar pieces by other composers). In that case, the separate voices of the piece are usually clearly indicated by being confined to particular staffs and notated with either upward or downward stems."


Voice Seperation - Sinf_13.mid
File Help
Open... Save As Midi
University of Southern California || 2003 || All Rights Reserved || xiaodanw@usc.edu

(a) main screen showing segmentation and voice count


Voice Seperation - Sinf_13.mid
File Help
Open... Save As Midi Ground Truth Analyzed by VoSA
15
16
17
21 22
23 24 25
26
University of Southern California || 2003-2004 || All Rights Reserved || xiaodanw@usc.edu

(b) the error locator screen showing voice assignments and erroneous connections (X)

Fig. 8. Screenshots of VoSA, the Voice Separation Analyzer

To facilitate evaluation of the voice separation procedure, we first need the ground truth, the correct assignment. An advantage of using Bach's fugues and his two- and three-part inventions is that many MIDI renditions of these pieces exist that have been sequenced such that each voice is in a separate track. For comparison against our results, we use such track separated MIDI files. The fugues were obtained from the MuseData repository, *www.musedata.org*, and the two- and three-part inventions from The Midi Archive at *archive.cs.uu.nl/pub/MIDI*. We used the scores from Virtual Sheet Music, *www.virtualsheetmusic.com*, for checking the voice assignments manually.

4.2 Evaluation Method

We use three main statistics to quantify the performance of the algorithm, namely, the *average fragment consistency*, the *correct fragment connection* rate and the *average voice consistency*. The evaluation process in VoSA records all the errors in the results and shows them visually as demonstrated in Figure 8(b). The GUI in VoSA allows the user to compare the voice assignments to the ground truth.

The *average fragment consistency* measures the overall percentage consistency over all fragments. A fragment is considered consistent if all notes in the fragment belong to the same voice. The percentage consistency of a fragment is the highest proportion of notes assigned to the same voice. This number shows the accuracy of the segmentation and fragment generation procedure. Formally, if V is the set of all voice indices, F the set of all fragments and $vN(note)$ the true voice assignment for *note*, then the percentage consistency of fragment f is defined as:

$$FC(f) = \frac{100}{\|f\|} \max_{v \in V} \{\| \, note \text{ in } f : vN(note) = v \|\},$$

where $\|f\|$ represents the cardinality of f, the number of notes in fragment f. The average fragment consistency is given by:

$$AFC = \frac{1}{\|F\|} \sum_{f \in F} FC(f). \tag{1}$$

The *correct fragment connection* rate measures the proportion of connections that are correctly assigned. The correctness of each connection is evaluated by comparing it to the ground truth obtained a track-separated MIDI file as described in Section 4.1. To describe the mathematical formula for this quantity, we first define C to be the set of all pairs of connected fragments, $\{(f, g) : f, g \in F$ and f is connected to $g\}$ and $vF(f)$ to be the true voice assignment for fragment f. In the case of 100% fragment consistency, $vF(f)$ is the true voice assignment of all notes in fragment f. When a fragment has less than 100% consistency, $vF(f)$ is the voice to which the majority of the notes in f belong. More formally, $vF(f) = \arg\max_{v \in V} \{\| \, note \text{ in } f : vN(note) = v \|\}$. The correct fragment connection rate is then given by the equation:

$$CFC = \frac{100}{\|C\|}\|\{(f,g) \in C : vF(f) = vF(g)\}\|. \tag{2}$$

Finally, the *average voice consistency* measures how well the notes in the piece have been properly assigned to their appropriate voices. This quantity measures, on average, the proportion of notes from the same voice that have been assigned by the algorithm to the same voice. Again, we begin with two definitions: let $vA(note)$ be the algorithm-assigned voice for *note* and $S(v)$ be the set of notes assigned to voice v, $\{note : vA(note) = v\}$. The voice consistency is defined as

$$VC(v) = \frac{100}{\|S(v)\|}\max_{u \in V}\{\|note \in S(v) : vN(note) = u\|\},$$

and the average voice consistency is given by:

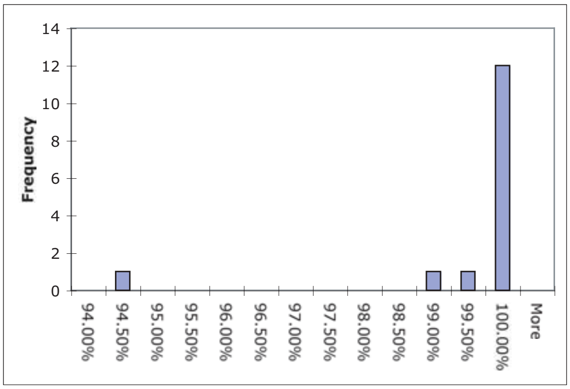
$$AVC = \frac{1}{\|V\|}\sum_{v \in V} VC(v). \tag{3}$$

4.3 Results

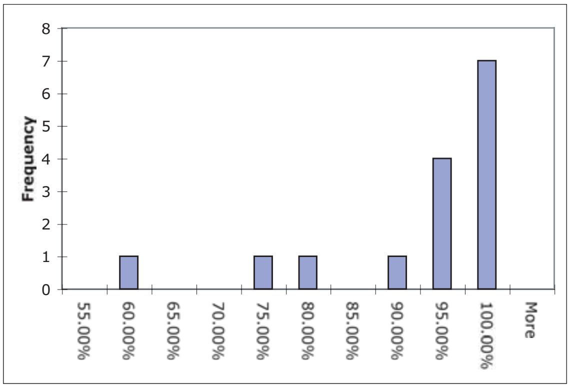
The contig mapping algorithm was tested on the 15 *Two-Part Inventions* (BWV 772-786), the 15 *Three-Part Inventions* (BWV 787-801) and the 48 Fugues from the *Well-Tempered Clavier* (BWV 846-893) by Johann Sebastian Bach (1685-1750). For each test sample, we used a quantization threshold of 30ms to pre-process the MIDI data before separating the voices using the contig mapping algorithm. We then evaluated the average fragment consistency (AFC), the correct fragment connection rate (CFC) and the average voice consistency (AVC) of the voice separation result. The distributions of these values for each test set – Two- and Three-Part Inventions and Fugues – are summarized in Figures 9, 10 and 11 respectively. The summary statistics are reported in Table 1.

The overall average fragment consistency (AFC) for the test corpus was 99.75%, that is to say, all notes in the same fragment are almost certain to be from the same voice. The overall correct fragment connection (CFC) rate was 94.50% indicating that the likelihood of connecting each fragment correctly to its contiguous strand is high. And, the overall average voice consistency (AVC) was 88.98%. Recall that this number reflects the proportion of notes in the same stream that were correctly assigned to the same voice by the algorithm. This number is lower than the AFC or CFC because each incorrect connection can result in a severe loss of voice consistency.

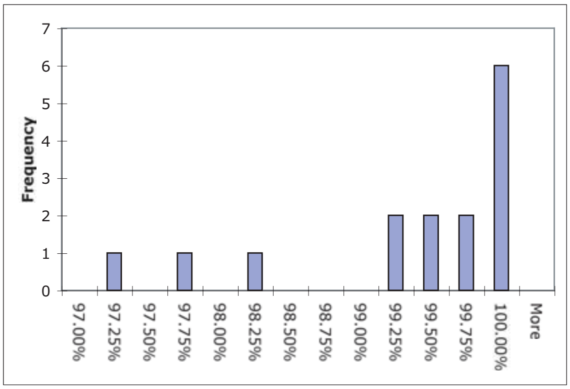
In general, higher average fragment sizes are correlated with higher average voice consistency numbers. This is not surprising considering that the average fragment consistency is extremely high. We found three possible sources for error in the contig mapping approach. The connection policy minimizes pitch distance. Even though this is generally the case, sometimes the shortest distance connection does not produce the correct result. On rare occasions, voices do cross, producing connection distances that are not minimal. Unintentional gaps between notes in the MIDI file that are not properly quantized can also lead to higher rates of error.



(a) average fragment consistency histogram (average AFC = 99.46%)

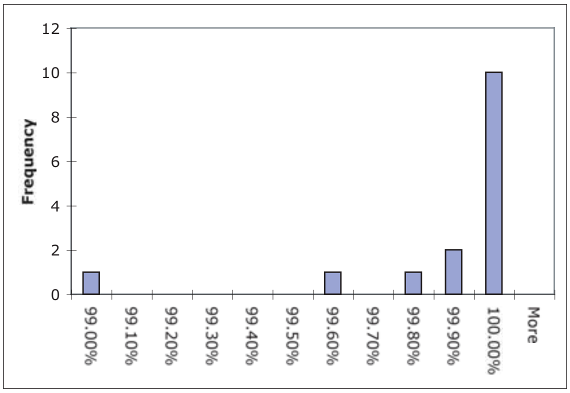


(b) average correct fragment connection histogram (average CFC = 91.47%)

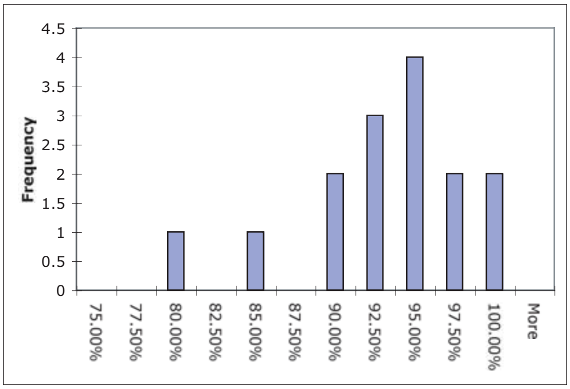


(c) average voice consistency histogram (average AVC = 99.29%)

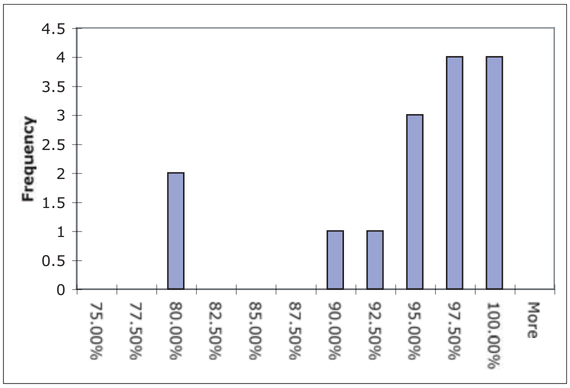
Fig. 9. Voice separation results for Bach's Two-Part Inventions.



(a) average fragment consistency histogram (average AFC = 99.80%)

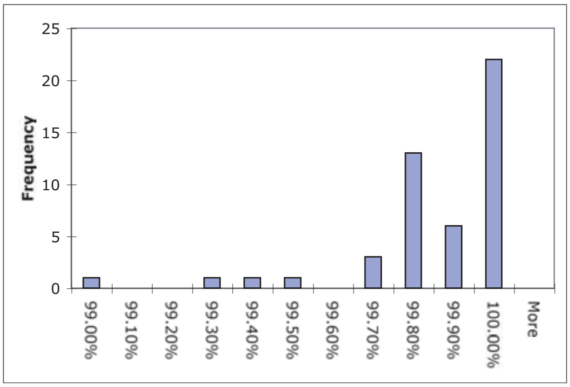


(b) average correct fragment connection histogram (average CFC = 92.27%)

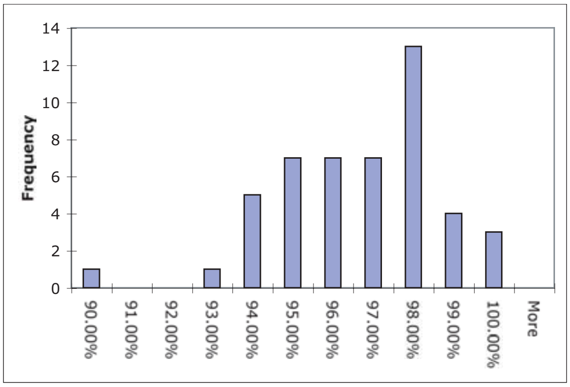


(c) average voice consistency histogram (average AVC = 93.35%)

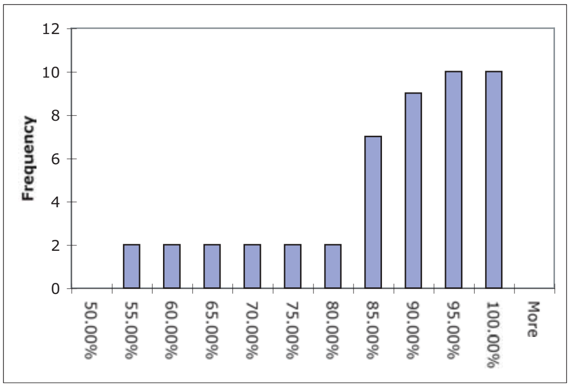
Fig. 10. Voice separation results for Bach's Three-Part Inventions.



(a) average fragment consistency histogram (average AFC = 99.83%)



(b) average correct fragment connection histogram (average CFC = 96.15%)



(c) average voice consistency histogram (average AVC = 84.39%)

Fig. 11. Voice separation results for Bach's 48 Fugues from the *Well-Tempered Clavier*.

Table 1. Summary statistics (average numbers) for voice separation experiments

MIDI input	no. of fragments per piece	average fragment size	no. of contigs per piece	average AFC (%)	average CFC (%)	average AVC (%)
Two-Part Inventions	46.67	18.26	32.60	99.46	91.47	99.29
Three-Part Inventions	194.67	4.28	82.33	99.80	92.27	93.35
WTC Fugues	581.81	3.05	226.50	99.83	96.15	84.39
OVERALL	404.45	6.21	161.49	99.75	94.50	88.98

5 Conclusions and Future Work

In this paper, we described a contig mapping approach to voice separation and three metrics for evaluating its voice separation results. The algorithm has been implemented in a voice separation analyzer application software called VoSA. We used VoSA to compute and analyze the voice separation results when the algorithm is applied to Bach's Two- and Three-Part Inventions and Fugues. Our experiments and evaluations are the first of this scope for the testing of a voice separation algorithm. The overall statistics are promising, showing that the contig mapping approach presents a computationally viable and highly accurate solution to the voice separation problem. Future work includes the testing of the algorithm on a larger polyphonic corpus, and extending the method to homophonic music.

6 Acknowledgements

We acknowledge the kind assistance of Fabrizio Ferrari and Laura Caldera for giving us access to Virtual Sheet Music's repository of classical sheet music.

The research has been funded in part by the Integrated Media Systems Center, a National Science Foundation Engineering Research Center, Cooperative Agreement No. EEC-9529152, and by a National Science Foundation Information Technology Research Grant No. ITR-0219912. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the National Science Foundation.

References

1. Bregman, A.: Auditory Scene Analysis: The Perceptual Organization of Sound. The MIT Press, Cambridge Massachusetts (1990) 417–442
2. Cambouropoulos, E.: From MIDI to Traditional Musical Notation. In Proceedings of the AAAI Workshop on Artificial Intelligence and Music: Towards Formal Models for Composition, Performance and Analysis, July 30 - Aug 3, Austin, Texas (2000)

3. Cambouropoulos, E.: Pitch Spelling: A Computational Model. Music Perception. **20**(4) (2003) 411–429
4. Chew, E., Chen, Y.-C.: Determining Context-Defining Windows: Pitch Spelling Using the Spiral Array. In Proceedings of the 4th International Conference on Music Information Retrieval. (2003)
5. Deutsch, D.: Two-channel Listening to Musical Scales. Journal of the Acoustical Society of America **57** (1975) 1156–1160
6. Goebl, W.: Melody Lead in Piano Performance: Expressive Device or Artifact? Journal of the Acoustical Society of America **110**(1) (2001) 563–572
7. Huron, D.: Tone and Voice: A Derivation of the Rules of Voice-leading from Perceptual Principles. Music Perception. **19**(1) (2001) 1–64
8. Kilian, J., Hoos, H.: Voice Separation - A Local Optimization Approach. In Proceedings of the 3rd International Conference on Music Information Retrieval. (2002) 39–46
9. Lemström, K., Tarhio, J.: Detecting monophonic patterns within polyphonic sources. In Content-Based Multimedia Information Access Conference Proceedings (RIAO 2000), Paris (2000) 1251–1279
10. Meredith, D.: Pitch Spelling Algorithms. In Proceedings of the Fifth Triennial ESCOM Conference. Hanover University of Music and Drama, Germany (2003) 204–207
11. Temperley, D.: The Cognition of Basic Musical Structures. The MIT Press, Cambridge Massachusetts (2001) 85–114

An Auditory Model Based Approach for Melody Detection in Polyphonic Musical Recordings

Rui Pedro Paiva, Teresa Mendes, and Amílcar Cardoso

CISUC – Center for Informatics and Systems of the University of Coimbra,
Department of Informatics Engineering, University of Coimbra (Polo II),
P3030 Coimbra, Portugal
{ruipedro, tmendes, amilcar}@dei.uc.pt

Abstract. We present a method for melody detection in polyphonic musical signals based on a model of the human auditory system. First, a set of pitch candidates is obtained for each frame, based on the output of an ear model and periodicity detection using correlograms. Trajectories of the most salient pitches are then constructed. Next, note candidates are obtained by trajectory segmentation (in terms of frequency and pitch salience variations). Too short, low-salience and harmonically-related notes are then eliminated. Finally, the melody is extracted by selecting the most important notes at each time, based on their pitch salience. We tested our method with excerpts from 12 songs encompassing several genres. In the songs where the solo stands out clearly, most of the melody notes were successfully detected. However, for songs where the melody is not that salient, the algorithm was not very accurate. Nevertheless, the followed approach seems promising.

1 Introduction

As a result of recent technological innovations, there has been a tremendous growth in the Electronic Music Distribution (EMD) industry. Factors like the widespread access to the Internet, bandwidth increasing in domestic accesses or the generalized use of compact audio formats with CD or near CD quality, such as mp3, have given a great contribution to that boom. Presently, it is expected that the number of digital music archives, as well as their dimension, grow significantly in the near future, both in terms of music database size and in number of genres covered.

However, any large music database, or, generically speaking, any multimedia database, is only really useful if users can find what they are looking for in an efficient manner. Today, whether it is the case of a digital music library, the Internet or any music database, search and retrieval is carried out mostly in a textual manner, based on categories such as author, title or genre. This approach leads to a certain number of difficulties, namely in what concerns database search in a transparent and intuitive way. Therefore, in order to overcome the limitations described, research is being conducted in an emergent and promising field called Music Information Retrieval (MIR).

Query-by-humming (QBH) [1, 4, 6] is a particularly intuitive way of searching for a musical piece, since melody humming is a very natural habit of humans. Therefore, several technologies have been developed that aim to permit such function. However, presently, this work is being carried out only in the MIDI domain, which places important usability questions. In fact, usually we look for recorded songs, which can be obtained from CDs or are stored in audio formats such as mp3. Additionally, in our opinion, looking for musical pieces in the MIDI format is an easier problem, since there the melody is very often available in a separate channel. The main issues are, then, to extract the notes from the hummed query (a well-known monophonic pitch[1] extraction problem,) and to match the query to the melody (an information retrieval problem). However, when the melody is not explicitly separated, the problem is not so easy since it must be extracted somehow. Several algorithms have been proposed to detect the melody in MIDI files, e.g. [21]. Anyway, we still maintain the opinion that dealing with audio files is a more challenging task since in MIDI, all the notes, as well as their timings, are already known, which simplifies the extraction of melody even when it is not directly available.

Querying "real-world" polyphonic recorded musical pieces requires that some sort of melody representation be extracted beforehand, which creates many more difficulties. Polyphonic musical signals can be converted to symbolic formats either manually or automatically. Manual conversion requires, obviously, a tremendous amount of man-work and specialized skills. On the other hand, analyzing polyphonic musical waveforms is a rather complex task, since we can have many different types of instruments playing at the same time, whose spectra interfere severely with each other. This fact makes it very complicated to separate the different sound sources.

Source separation is a major concern for polyphonic music analysis and automatic music transcription systems and has no general solution yet. One way to approach this problem is to build computer models that emulate human auditory processing. It is generally agreed that the human brain processes auditory information in a way called "auditory scene analysis" [3]. As an attempt to replicate human behavior, some work has been carried out aiming to develop computational auditory scene analysis systems. The results obtained are not very accurate yet and are only acceptable for simpler or well-constrained problems. Namely, Ellis [5] tries to analyze a sound waveform by means of competitive theories, where each of them proposes a combination of sounds that might have produced the resulting sound. Sound source models are used as a basis for the proposed method. Bello *et al* [2] and Martin [13] have used computational blackboard systems for simple automatic music transcription. The blackboard system is composed of a global database, where hypotheses are proposed and developed, a scheduler that determines how hypotheses are developed, and knowledge sources, corresponding to experts. Scheirer [15] proposes a model based on perceptual issues, using dynamic clustering of comodulation data. In contrast to the other systems referred, this model is designed for analysis of complex music. Klapuri [10] proposed a method for multi-pitch estimation where the musical signal is analyzed at separate frequency bands. Namely, 18 logarithmic distributed bands from 50 Hz to 6 kHz are used. Then at each band, a fundamental frequency likelihood vector is calculated. Finally, the results from each band are combined to yield global pitch likelihoods. They report results that outperform the average of ten trained musi-

[1] In this paper, we use the term pitch indistinctly of fundamental frequency, though the former is a perceptual variable, whereas the latter is a physical one.

cians. Other models impose constraints in the number of instruments present or the harmonic interaction between them, as referred in [7].

Melody detection can be seen as a sub-problem of polyphonic pitch detection and source separation, where the aim is to detect the main melodic line, regardless of the other sources present. This requires the detection of the dominant notes at each time, not the whole set of notes present. For instance, when we hear a pop song, we have vocals, guitar, bass, percussion and so forth. Yet, in spite of all that information, our brains still can retain the main melodic line.

Only little work has been carried out in the particular problem of melody detection in "real-world" songs. One interesting approach is the one followed by Goto [7]. The author uses a probabilistic model for the detection of melody and bass lines. The sound wave is first band-pass filtered and then a probability density function (*pdf*) is computed for each signal component. The *pdf*s are generated from a weighted-mixture of tone models of all possible fundamental frequencies. The more dominant a model is in the PDF, the more likely the fundamental frequency belongs to that model. The author compared the dominant frequencies detected with hand-labeled marked notes and reports an average rate of 88.4% for the melodic pitch line.

Song *et al* [20] use a different approach, based on the fact that there is no single method that is both accurate and generic. They argue that their method is more pragmatic when the final goal is QBH: instead of trying to extract the melody, they use a mid-level melody representation, which consists of a sequence of audio segments where each segment contains a set of note candidates. Then, they use a variation of dynamic programming for matching the query with the melody mid-level representation.

In this paper, we describe a multi-stage method for melody detection, based on a model of the human auditory system [17]. Since source separation is a complex problem for which no general and accurate solutions exist yet, we try to evaluate what we think is a more pragmatic approach, at least for the time being: we extract the melody based on the assumption that it generally clearly stands out of the background. In short, the method works as follows. The sound wave is first divided into frames where stationarity can be assumed. For each frame, we get a set of pitch candidates, based on the output of an ear model and periodicity detection using correlograms. We determine pitch candidates by finding relevant peaks in a summary correlogram, where peak amplitude gives information regarding its salience. Then, we create pitch trajectories, based on frequency proximity. After trajectory creation, note candidates are obtained by trajectory segmentation and elimination. Short-duration, low-salience and harmonically-related notes are then eliminated. Finally, the melody is extracted by selecting the most important notes at each time, based on their pitch saliences.

We tested our system on excerpts of 12 songs, encompassing several different genres. The obtained notes were then compared with the correct ones, previously hand-labeled. In the songs where the solo stands out clearly, most of the melody notes were successfully detected. However, for songs where the melody is not that salient, the algorithm was not so accurate. Yet, we could say that the obtained results are encouraging.

The following sections describe the work carried out in this paper. Section 2 describes the melody detection method. In Section 3, experimental results are presented and evaluated. Finally, in Section 4, conclusions are drawn and possible directions for future work are pointed out.

2 Melody Detection Method

Our melody detection algorithm is composed of five modules, illustrated in Fig. 1.

The first module, multi-pitch detection (MPD), receives a raw polyphonic musical signal and returns a set of pitch candidates and their respective saliences. Then, pitch trajectories are created based on frequency proximity, in the multi-pitch trajectory construction (MPTC) module.

The resulting trajectories are then segmented, based on frequency and pitch salience variations, leading to an initial set of candidate notes. Since many of the obtained notes are irrelevant in a melody extraction context, short-duration, low-salience and harmonically-related notes are eliminated. Finally, the notes comprising the detected melody are extracted by selecting the most salient notes at each time.

For the sake of visualization simplicity, we will illustrate the method with a simple example: a monophonic saxophone riff.

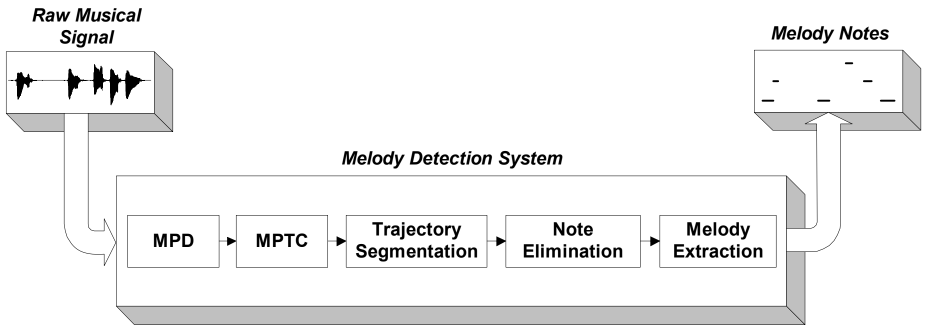


Fig. 1. Melody detection system overview

2.1 Multi-pitch Detection

In the first stage of the algorithm, the objective is to capture a set of candidate pitches, which constitute the basis of possible future notes. The MPD algorithm receives as input a raw musical signal (monaural, sampling frequency f_s = 22050 Hz, 16 bits quantization) and outputs a set of pitch candidates and respective saliences.

Our goal is to obtain pitch candidates at each time instant. Since we cannot define instantaneous time in a computational model, we have to define some sort of time granularity. Therefore, we select a small enough time window and perform sound wave analysis in a frame-based way. We use a 20 ms frame length, which constitutes a good trade-off between time and frequency resolution: it is small enough for the assumption of signal stationarity and large enough for accurate detection of pitches above 100 Hz. The corresponding number of samples per frame is $N = 441$. In order to allow for a smooth transition between frames, 50% overlap is employed.

After dividing the musical signal into frames, we perform an auditory model based analysis of each frame, in order to detect the most salient pitches in each. This analysis comprises four stages, diagrammed in Fig. 2: i) conversion of the sound waveform into auditory nerve responses for each frequency channel, using a model of the ear, with particular emphasis on the cochlea (which creates an image called cochleagram); ii) detection of the main periodicities in each frequency channel using autocorrelation (which produces an image called correlogram); iii) detection of the global periodicities in the sound waveform by calculation of a summary correlogram; and iv) detection of the pitch candidates in the frame by looking for the most salient peaks in the summary correlogram.

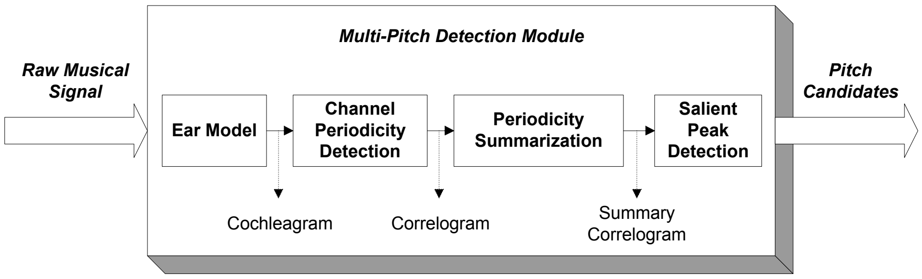


Fig. 2. Multi-pitch detection module

Ear Model. In the first stage of the multi-pitch detection system, a model of the ear is implemented, which aims to mimic the tasks carried out by the outer, middle and, particularly, the inner ear in the first stages of auditory processing. In the inner ear, the cochlea encodes information in the sound wave into a multi-channel representation of auditory nerve firing patterns. The output of the cochlear model is a two-dimensional representation of a sound waveform that permits its visualization as a time-frequency image. In this image, called "cochleagram", each line contains information regarding auditory nerve responses for the corresponding cochlear, or frequency, channel. A good review of the tasks carried out in the cochlea and auditory nerve can be found in [8; 9].

In the present work, we use the ear model proposed by Richard Lyon [11] and implemented by Malcolm Slaney [17; 18], with some minor adaptations. Below, we give a short description of Lyon's model. For a thorough analysis, we refer the reader to [17].

The model implements three main tasks: filtering, detection and compression. First, a cascade of second-order filters models sound propagation down the basilar membrane, which acts as a frequency analyzer. Therefore, each filter corresponds to a cochlear channel that best responds to a particular frequency range. Furthermore, front filters are also implemented, which constitute a simple model of the responses of the outer and middle ears. In the present implementation, with a sampling frequency of 22050 Hz, 96 cochlear filters are used. Fig. 3 depicts every 5th filter response, using a logarithmic frequency axis. This figure was created using Slaney's Auditory Toolbox [18]. As for model parameterization, we use the default parameters proposed by Slaney, namely a filter Q of 8 and a step factor of 0.25 (which determines the amount of filter overlap).

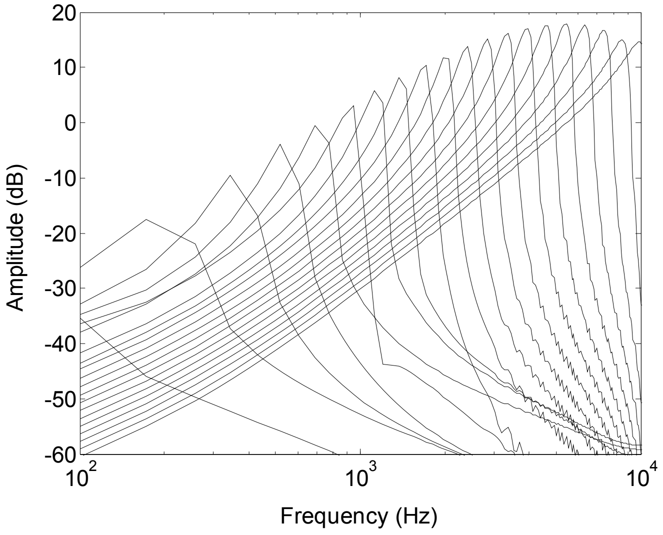


Fig. 3. Frequency response of cochlear filters

After filtering, the movements of the basilar membrane are converted into auditory nerve responses. Since inner hair cells only respond to movement in one direction, an array of half-wave rectifiers is employed to detect the output of each second order filter. This is a simple model of detection that does not account, for instance, for saturation effects.

Finally, four stages of automatic gain control compress the dynamic range of the input into a limited level that the auditory nerve can deal with. The automatic gain control is, in fact, a model of ear's adaptation: the response to a constant stimulus is first large and then, as the auditory system adapts to the stimulus, the response becomes smaller. Regarding parameterization, we use once again the parameters proposed by Slaney [18], namely target values of 0.0032, 0.0016, 0.0008 and 0.0004 and time constants of 640, 160, 40 and 10 ms for the first, second, third and fourth stages of automatic gain control, respectively.

Fig. 4 presents a cochleagram for our monophonic saxophone riff example: here, the harmonics of the sound waveform are clearly visible by the horizontal striations. Recall that higher channels correspond to lower frequencies. This picture has a limited time resolution, due to displaying purposes. However, the inner hair cells in the cochlea are extremely sensitive to the time structure of each component of the sound. Thus, a view of the cochleagram for a 20 ms' time slice is presented in Fig. 5b. In this figure, the harmonics are not so clear but a more precise image of auditory nerve firing responses in each channel is obtained.

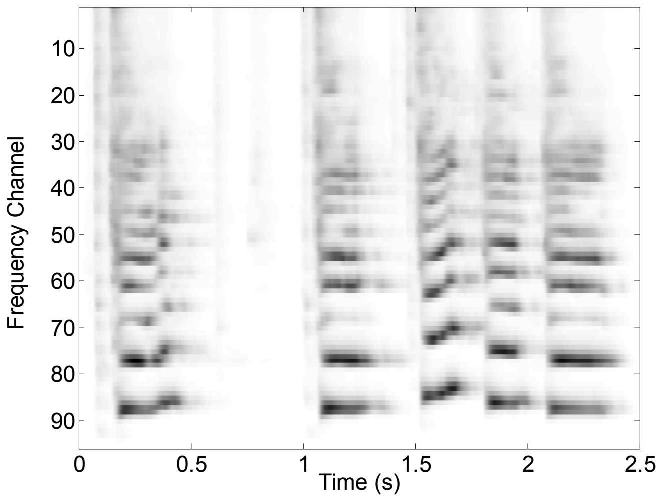


Fig. 4. Cochleagram of a 2.5s' saxophone riff

Channel Periodicity Detection. After computing the auditory nerve firing responses for each frequency channel, the main periodicities in the sound wave are detected. Here, this is accomplished by computation of the auto-correlation function (ACF) in each channel, resulting in a two-dimensional image of the sound signal, where the horizontal axis represents correlation lag and the vertical axis represents frequency. This image is called "correlogram" which means, literally, "picture of correlations" [17]. Each line of the correlogram contains information regarding the salience of the periodicities found for a given frequency channel. Like the cochleagram, the correlogram activity is measured by pixel intensity in the image.

The main objective of the correlogram is to summarize the temporal activity at the output of the cochlea [17]. In fact, many sounds, and particularly musical sounds, are periodic in time, or at least pseudo-periodic. The correlogram is, then, a powerful tool for detecting and visualizing the referred periodicities. As a result, all channels will show peaks at the horizontal positions corresponding to correlation lags that match the periods of repetition present in the signal.

Slaney [17] argues that the correlogram is biologically plausible. In fact, despite the separation of sound into broad cochlear channels, the temporal properties of the original signal are still kept. It is likely that the brain measures periodicities using a neural delay line, a case that is supported by the cross-correlator structures found in the brains of owls and cats. Furthermore, the detection of periodicities is also inspired by the well-known "timing theory" of auditory nerve firing.

In terms of computer implementation, here, the periodicities in the cochleagram are obtained by computing the short-time ACF of the neural firing responses in each cochlear channel for a particular time window. As was referred previously, the sound wave must be divided into frames where stationarity can be assumed. This is equivalent to multiplying the signal by a sliding rectangular window. However, in order to smooth the correlation, a Hamming window is used instead. In order to improve efficiency, the ACF in each window is implemented via the fast Fourier Transform (FFT) algorithm, which is equivalent to performing circular auto-correlation [19].

It is common to normalize the ACF so that its value at zero lag is equal to one, in order to reduce is dynamic range. However, this procedure eliminates any indication of the relative power in different cochlear channels. Therefore, the correlations are partially normalized by the square root of the power [18]. In this way, its dynamic range becomes comparable to the one of the cochleagram, keeping the relative powers between channels [17]. An example of a 20 ms' correlogram frame for the saxophone riff is presented in Fig. 5b. This picture shows the utility of correlograms for the analysis of periodic signals: there are clear vertical lines at particular auto-correlation lags, indicating instants when a large number of cochlear channels fire with the same period. This in turn is a clear indication of the pitch periods present in the signal.

Periodicity Summarization. As we referred above, the vertical lines across several cochlear channels show evidence of pitch. Therefore, a summary correlogram (SC) is computed by summing the ACFs across all channels at each time lag. This measures the likelihood that a periodicity corresponding to a particular time lag is present in the sound waveform.

Generally, the SC looks very noise, with many spurious peaks that make peak detection more difficult. Therefore, peak enhancing should be carried out. Here, we do not follow Slaney's approach [17]. Instead, we perform peak enhancing by smoothing (where we apply a low-pass filter) and squaring (in order to emphasize the peaks). Since the correlograms in the previous stage are all non-negative, the SC will also be non-negative and so does not need to be rectified, as would normally be the case. Moreover, unlike Slaney who normalizes the summary correlogram in each frame (dividing it by the value at zero lag) [18], we use its exact values, since they are useful for trajectory segmentation, as will be explained in Section 2.3.

An example of a summary correlogram is presented in Fig. 5d, where the determined pitch candidates are marked, as will be described below.

Salient Peak Detection. The final stage of the multi-pitch detection module consists of finding a set of pitch candidates based on the most salient peaks in the summary correlogram. To accomplish this task, we first look for all peaks in the SC, excluding the one at zero lag, and obtain their respective saliences, i.e., their amplitudes. Then, we eliminate all peaks that are not salient enough. To accomplish this task, we find the highest peak salience, *maxPeakSal*, and determine the minimum allowed peak salience, *minPeakSal*, using the minimum salience ratio parameter, *minSalRatio*.

The detection of the main periodicities for our example is illustrated in Fig. 5d, where the most salient peaks, i.e., pitch candidates, are marked. The frequencies for the pitch candidates are then obtained by inverting the periods corresponding to the found peaks. Finally, the pitch saliences in all frames are normalized to the [0; 100] interval, for comparison in the following stages of the melody detection system.

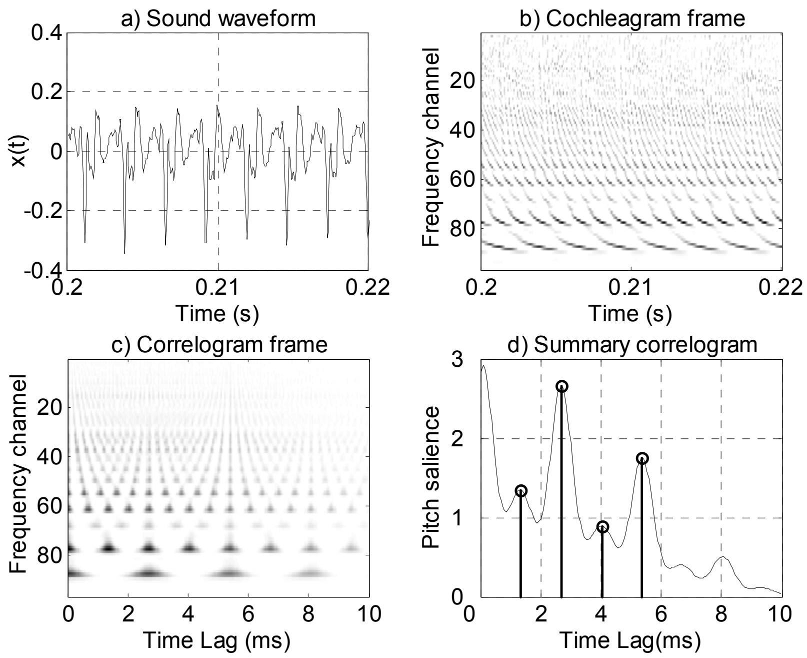


Fig. 5. Illustration of the four stages of the MPD algorithm

The four stages of the MPD algorithm are illustrated in Fig. 5: panel a) presents a 20 ms frame of the saxophone riff; panels b) and c) depict the corresponding cochleagram and correlogram images, respectively; and panel d) shows the summary correlogram, where the candidate pitch periods are marked.

At this point, the motivation for extracting multiple pitches when we are only interested in the melodic line deserves a better explanation. Actually, extracting a single pitch would be both easier and more intuitive. However, since we are performing pitch detection in a polyphonic context, it often happens that the pitch corresponding to the melody is not the most salient one in every frame. In fact, peaks corresponding to the periodicities of simultaneous notes may compete in the salience curve and be alternately selected as the maximum. Therefore, selecting several pitch candidates at this stage allows for the detection of lower-salience melody notes, which might not be captured if only a single pitch was extracted. In this way, it is possible to keep track of the global temporal continuity of each peak. We performed some experiments, to be reported in a future publication, which confirmed our assumption. The issue of note salience in a mixture of simultaneous notes is then dealt with in the following stages of the melody detection system.

The methodology for multi-pitch detection is summarized in Algorithm 1. Parameter definition is presented in Table 1. The parameters for the cochleagram are not presented, since we used the default values defined by Slaney [18], as referred above.

Algorithm 1. Multi-pitch detection

```
1. Compute the cochleagram for each time frame
   1.1. Apply Lyon's cochlear model
2. Compute the correlogram for each time frame
   2.1. Multiply each line of the cochleagram frame by
        a Hamming window
   2.2. Determine the ACF function for each channel via
        the FFT
   2.3. Normalize the ACF
3. Compute the summary correlogram for the each time
   frame
   3.1. Sum the ACF across all channels
   3.2. Enhance peaks: squaring + low-pass filtering
4. Detect salient peaks in the summary correlogram
   4.1. Determine minimum allowed peak value (salience)
        - $maxPeakSal \leftarrow$ maximum peak value
        - $minPeakSal \leftarrow maxPeakSal \times minSalRatio$
   4.2. Eliminate pitches with low salience
        4.2.1. If peak salience < $minPeakSal$, eliminate
               peak
   4.3. Convert pitch periods to frequencies
5. Normalize pitch saliences in all frames to the
   [0; 100] interval
6. Return pitch frequencies and saliences for all
   frames.
```

Table 1. MPD parameters

Parameter Name	Parameter Value
frame length	20 ms
frame overlap	50%
minSalRatio	0.2

Unlike automatic music transcription systems, this algorithm does not deal with the well known and complex “octave problem”. In fact, at this stage it is not important to analyze if a given pitch candidate corresponds to a real note or appears as a ghost note, whose fundamental frequency is a harmonic of some real note, a few octaves above. Some of the ghost notes will be eliminated already at this stage based on the pitch salience threshold, whereas others will be eliminated in the following stages of the melody detection algorithm.

2.2 Trajectory Construction

The second stage of the melody detection algorithm aims at creating a set of pitch trajectories, formed by connecting consecutive pitch candidates with similar frequencies. The idea is to find regions of stable pitches, which indicate the presence of musical notes. The MPTC algorithm receives as input a set of pitch candidates, characterized by their frequencies and saliences, and outputs a set of pitch trajectories, which constitute the basis of the final melody notes.

We follow rather closely Serra's peak continuation algorithm [16], who based himself in McAulay and Quatieri's work [12]. However, since we have a limited set of pitch candidates per frame, our algorithm is much lighter. In fact, Serra looks for regions of stable sinusoids in the signal's spectrum, which leads to a trajectory for each harmonic component found. Therefore, a high number of trajectories have to be processed, which makes the algorithm much heavier, though the basic idea is the same. Another difference is that we first quantize frequencies to the closest MIDI note. We found that peak continuation based on MIDI note numbers allows for a more robust trajectory build up. One reason for this seems to come from the fact that the location of peaks oscillates somewhat due to interference from other sources in the sound mixture. Furthermore, the representation of notes using MIDI numbers simplifies an eventual representation of the sound waveform in MIDI format (e.g., for generation of a MIDI file).

This algorithm is based on the definition of a maximum frequency deviation (in semi-tones in our case) for continuing trajectories. We define a value of one semitone, motivated by the fact that some songs comprise glissando and vibrato regions, as well as by the frequency oscillations that may result from interference of other sources. Therefore, in this way, all these phenomena are kept within a common track, instead of being separated into a number of different trajectories, e.g., one trajectory for each note that a glissando may traverse. The drawback of allowing a larger frequency deviation is that a single trajectory can contain more than one note. This is the reason why we perform trajectory segmentation, in the next stage of the melody detection algorithm.

Also, we specify a maximum number of frames where a trajectory can be inactive, i.e., when no continuation peaks are found. If this number is exceeded, the trajectory is stopped. Here, we define a maximum of 5 inactive frames.

Finally, any trajectory must be longer than a minimum trajectory length. Therefore, all finished trajectories that are shorter then this threshold, are eliminated. The minimum trajectory length in our implementation was set to 9 frames.

We present a detailed description of the implemented algorithm in [14].

The result of the MPTC algorithm is illustrated in Fig. 6, for our saxophone riff example. There, we can see that some of the obtained trajectories comprise glissando regions. Also, some of the trajectories include more than one note and should, therefore, be segmented.

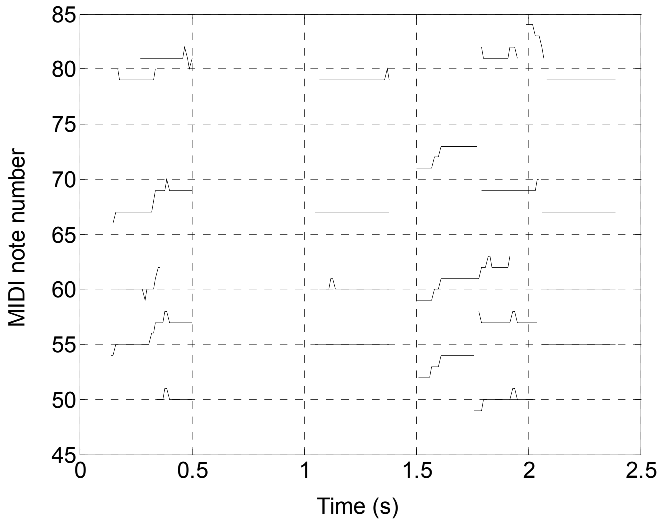


Fig. 6. Illustration of the MPTC algorithm

2.3 Trajectory Segmentation

As we mentioned previously, the trajectories that result from the MPTC algorithm may contain more than one note and, therefore, must be segmented. This is the task of the third stage of the melody detection method. The trajectory segmentation algorithm receives as input a set of trajectories of pitch candidates and outputs a set of segmented trajectories, i.e., note candidates.

Two types of segmentation have to be conducted. The most intuitive one is frequency segmentation, where the goal is to separate all the different frequency notes that are present in the same trajectory. The other one, pitch salience segmentation, aims at separating consecutive notes that have the same fundamental frequencies, which the MPTC algorithm may have interpreted as forming only one note. This requires segmentation based on salience minima, which mark the limits of each note. Here, it is important to say that the salience value depends on the evidence of pitch for that particular frequency, which is lower on the onsets and offsets. Consequently, the envelope of the salience curve is similar to an amplitude envelope: it grows at the note onset, has then a more steady region and decreases at the offset. Thus, notes can be segmented by detecting clear minima in the pitch salience curve.

As for frequency segmentation, the main idea is to find sufficiently long sequences of the same note number. Only then trajectories are segmented. When note transitions are found but the current note sequence is not long enough, i.e., larger than nine frames (as defined in the MPTC algorithm), the trajectory is not segmented, since it may correspond to the start of a glissando region. Furthermore, when we find short sequences delimited by the same note number, e.g., {70, 71, 71, 71, 71, 70}, these are interpreted as possible modulation regions, and so no segmentation takes place.

After frequency segmentation, the obtained candidate notes must be analyzed so as to check whether they should be further divided. In fact, there may be consecutive distinct notes at the same fundamental frequency that, erroneously, form a unique long note. In this situation, those notes must be segmented. In order to accomplish this task, salience segmentation takes place. The main idea is to find clear pitch salience minima that suggest the presence of more than one note, as referred before.

Finally, after all notes are segmented, their onset and offset times are adjusted. For each note, we get its maximum salience value and then define the onset as the first frame were the salience rises above 20% of the maximum salience found. The procedure is the same for the offsets, i.e., the note offset corresponds to the first frame where the salience rises above 20% of the maximum salience, starting from the end.

A detailed description of the implemented algorithm is presented in [14].

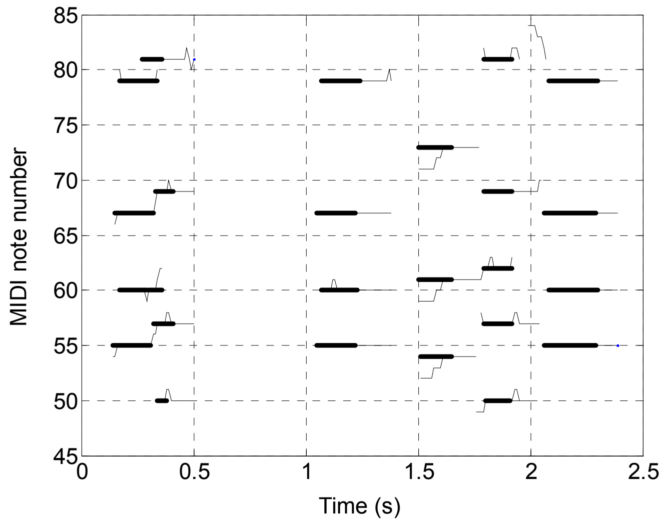


Fig. 7. Illustration of the trajectory segmentation algorithm

Fig. 7 illustrates trajectory segmentation, using the initial trajectories from the MPTC algorithm (Fig. 6). The obtained notes are depicted with thick lines. We can see that glissando and modulation regions are properly dealt with (check notes starting approximately at time 1.5s). Particularly, the transition that is observed in Fig. 4 around time 1.5 s, which corresponds to a glissando, was kept within the same note instead of being separated into the several notes it traverses. Furthermore, some trajectories are truncated as a consequence of the assumption for onset and offset detection.

2.4 Note Elimination

The objective of the fourth stage of the melody detection algorithm is to delete some of the note candidates, based on their saliences, durations and on the analysis of harmonic relations. The note elimination algorithm receives as input a set of note candidates and outputs a reduced set of notes, relevant for melody extraction.

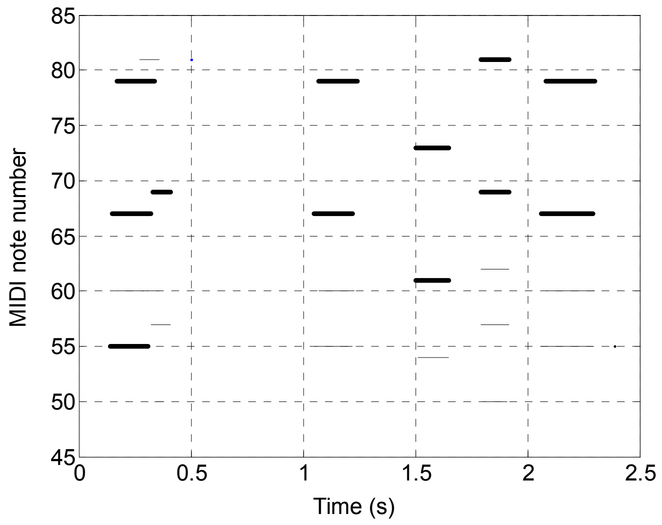


Fig. 8. Illustration of the note elimination algorithm

First, low-salience notes are deleted. A note is low-salience if its average salience is below 20 and if the number of frames whose salience is above that threshold is not enough, i.e., less than 5 frames. Next, all the notes that are too short, i.e., whose duration is below the minimum of 9 frames, defined in the MPTC algorithm, are also deleted. Finally, we look for harmonic relations between all notes, based on the fact that some of the obtained pitch candidates are sub-harmonics of real pitches in the sound wave. If two notes have approximately the same onset and offset times and are harmonically related, it is possible that the lower one is just a sub-harmonic of the higher one. Therefore, we compare their respective saliences in order to take a decision: if the salience of the lower note is less than 60% of the salience of the higher note, the lower one is eliminated. We describe this algorithm in greater detail in [14].

Fig. 8 illustrates note elimination, based on the note candidates of Fig. 7. The obtained notes are depicted with thick lines. It can be seen that many of the note candidates are eliminated at this point.

2.5 Melody Extraction

In the final stage of the present melody detection system, our goal is to obtain a final set of notes comprising the melody of the song under analysis. The melody extraction algorithm receives as input the set of notes returned by the note elimination algorithm and outputs the final melody notes.

This stage of the proposed system, being probably the most important one, is also the most difficult one to carry out. In fact, many aspects of auditory organization influence the perception of melody by humans, for instance in terms of the role played by the pitch, timbre and intensity content of the sound signal. In our approach, we do not attack the problem of source separation, as would normally be the case. Instead, we base our strategy on the assumption that the main melodic line often stands out in the mixture, as referred in the Section 1.

This algorithm starts by analyzing intersections between notes. The beginning and end of intersection regions is used to segment the sound signal, as illustrated in Fig. 9, where s_i stands for the i-th obtained segment.

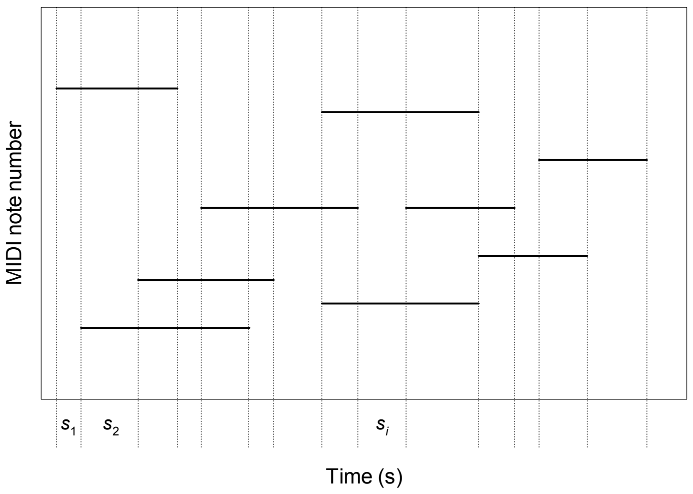


Fig. 9. Segmentation based on note intersection

Then, for each segment, we determine the three most salient notes, based on the average pitch salience of each note in each segment. Notes below MIDI note number 50 (146.83 Hz) are excluded. This procedure is motivated by the fact that the notes comprising the melody are, usually, in a middle frequency range.

Next, we eliminate all the notes that are not dominant, i.e., that are not in the three most salient notes for more than 2/3 of their total number of frames or do not have the highest salience for more than 9 frames. Finally, we do not allow any simultaneous notes. Therefore, for notes with approximate onsets and offsets we keep only the most salient one, then we eliminate notes included in larger duration notes and truncate notes that end after the next note starts (or vice-versa, depending on their respective saliences in the common segments).

Our melody extraction algorithm is described in detail in [14].

Fig. 10 illustrates melody extraction, based on the example in Fig. 8. The final melody notes are depicted with thick lines.

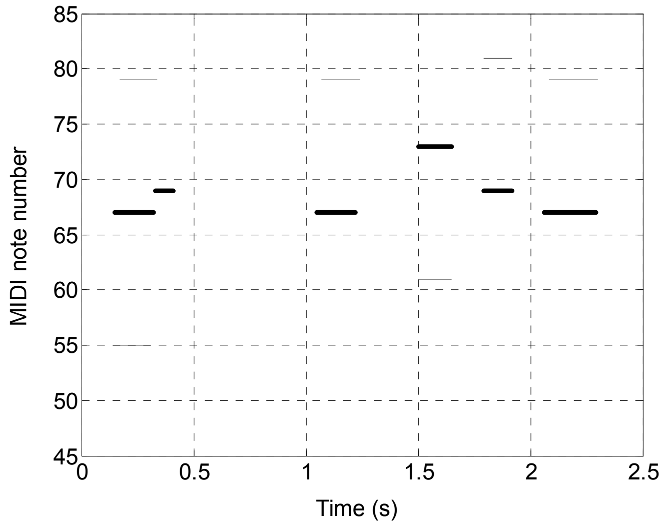


Fig. 10. Illustration of the melody extraction algorithm

3 Experimental Results

One difficulty regarding the evaluation of MIR systems results from the absence of standard test collections and benchmark problems. Therefore, we created our own test database, having care regarding its diversity and musical content. We collected excerpts of about 6 seconds from 12 songs, encompassing several different genres. The selected songs contain a solo (either vocal or instrumental) and accompaniment parts (guitar, bass, percussion, other vocals, etc.).

The obtained results are summarized in Table 2. There, “V” stands for vocals and “I” stands for instrumental. Fig. 11 shows an example of the results of the melody detection system for an excerpt of the song “Thank You”, by Dido.

In the example in Fig. 11 we can see that the correct notes (thick lines) match the obtained melody notes (thin continuous lines) in most of the cases. The undetected notes are marked with circles. As can be seen, two of the three missing notes were present in the notes obtained after elimination (dotted lines). One of the missing notes, approximately at time 5.8s, corresponds to erroneous trajectory segmentation. There are also other minor segmentation errors.

Table 2. Results of the melody detection system

Song Title	Genre	Solo Type	#Total Notes	#Correct Notes
Pachelbel's Kanon	Classical	I	16	8 (50%)
Handel's Hallelujah	Choral	V	15	n. r.
Enya – Only Time	Neo-Classical	V	11	10 (90,9%)
Dido – Thank You	Pop	V	16	13 (81.25%)
Ricky Martin – Private Emotion	Pop	V	10	6 (60%)
Avril Lavigne – Complicated	Pop/Rock	V	14	9 (64.3%)
Claudio Roditi – Rua Dona Margarida	Jazz / Easy	I	19	18 (94.7%)
Mambo Kings – Bella Maria de Mi Alma	Bolero	I	12	8 (75%)
Compay Segundo – Chan Chan	Latin	V	10	n. r.
Juan Luis Guerra – Palomita Blanca	Latin Rumba	V	10	8 (80%)
Battlefield Band – Snow on the Hills	Scottish Folk	I	26	20 (76,9%)
Saxophone riff	(monophonic)	I	6	6 (100%)

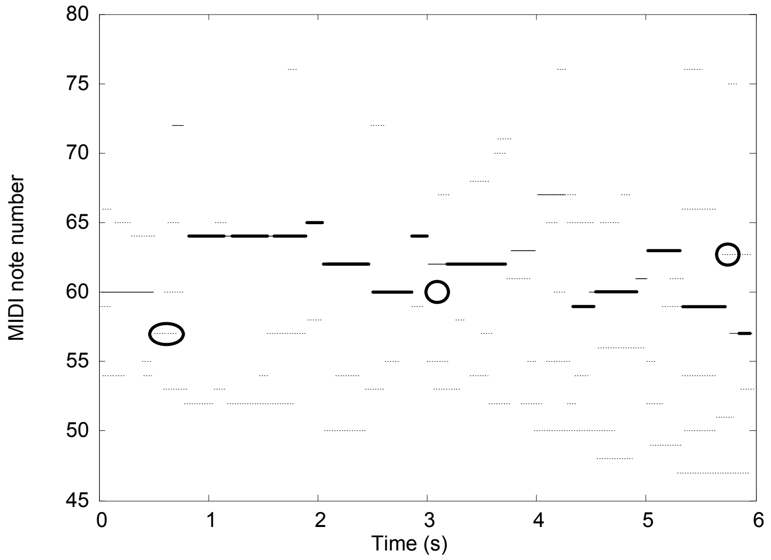


Fig. 11. Detected melody for "Dido - Thank You" excerpt

The detected melody notes were compared with the correct notes, previously hand-labeled. In the absence of the melody line, the system detected the dominant accompaniment part, since sound sources are not discriminated. This can be seen in Fig. 11, by the thin continuous lines. This is consistent with the way humans seem to memorize melodies: a mix of solo regions with accompaniment regions, in the absence of a solo. However, we decided to ignore the notes where the accompaniment part dominates, in the same way as Goto does [7]. In order to extract only the melody, we would need a means of separating notes according to their sources. The most intuitive, but complex, way to accomplish this task would be to use timbre models. Other possibilities would be to separate notes according to their frequency ranges, note intensity levels (since the intensity of a solo varies usually in a smooth way) or duration of notes (e.g., it is not likely that a short duration note in the middle of two long notes belongs to the same source as them).

In our test cases, we observed that some of the notes were erroneously segmented (too much or too little segmentation) and others were shorter than the original ones. This resulted from noise in both the frequency and salience sequences, as well as frequency deviations in the MPTC algorithm, which lead to excessive trajectory segmentation. The noise in the salience sequences results often from interference from other sources in the sound mixture, namely percussive instruments. One possible way to deal with this issue would be to smooth the frequency and salience sequences before segmentation. Another possibility would be to filter out percussive sounds from the mixture, which is a challenging task. We also observed a few semi-tone deviations (a small number of them). These errors resulted from the previous one and so should diminish after we deal with the problems coming from segmentation. We decided to ignore these small errors since our goal is to check whether a note is present or not, no matter in how many sub-notes the algorithm divides it. These errors can be reduced as was referred.

We can see that the algorithm could not find any reasonable melody in some excerpts ("#Correct Notes = n. r.: not reasonable"). However, in the cases where the melody stands clearly out of the background and percussion is not too intense, good results were achieved, which matches Goto's results [7]. In two of examples, the system achieved an accuracy close to 100% (only one missing note). Furthermore, the results obtained for pop/rock and rumba songs surprised us positively, since they have strong percussion (Juan Luis Guerra), as well as intense guitars (Ricky Martion) with distortion (Avril Lavigne).

It is also worthwhile to say that, in the tested examples, many of the missing notes were still present after the note elimination stage. This suggests that a more robust melody extraction module could lead to better results.

We also tested our system with a simple monophonic saxophone riff, as referred throughout this paper. In this example, the results were very good in terms of detection of glissandos, vibratos and note onsets and offsets. Consequently, we hope our system could be used as a robust monophonic pitch detection tool.

4 Conclusions

We have presented a system for melody detection in polyphonic musical signals. This is a main issue for MIR applications, such as QBH "real-world" music databases. The work conducted in this field is presently restricted to the MIDI domain, and so we guess we make an interesting contribution to the area, though our results were not satisfactory enough for real applications. However, the achieved results are encouraging, since we have not exploited the full potential of our approach yet. Furthermore, to our knowledge, only Goto [7] addresses the issue of melody detection in polyphonic music, but without trying to explicitly extract notes. Also, our system is reasonably simple and light, except for the multi-pitch detection module, due to cochlear modeling and auto-correlation computation.

Regarding future work, we plan to further work out some of the described limitations, namely devising a more robust algorithm for melody extraction. Additionally, we plan to apply some sort of pre-processing in order to filter out percussive sound components, which is a very demanding task. Therefore, we plan to evaluate the feasibility of Independent Component Analysis for source separation. The main idea would be to separate the solo and accompaniment parts (namely, percussive ones) and then detect the melody in the solo part using our proposed approach. Additionally, we plan to evaluate and compare other types of pitch detectors and to adapt some of the techniques proposed in the monophonic pitch extraction context.

5 Acknowledgements

This work was partially supported by the Portuguese Ministry of Science and Technology (MCT), under the program PRAXIS XXI. We also would like to thank Malcolm Slaney for making available the source code of his Auditory Toolbox.

References

1. Bainbridge, D., Nevill-Manning, C., Witten, I., Smith, L., McNab, R.: Towards a Digital Library of Popular Music. ACM International Conference on Digital Libraries (1999) 161-169
2. Bello, J. P., Monti, G., Sandler, M.: Techniques for Automatic Music Transcription. First International Symposium on Music Information Retrieval (2000)
3. Bregman, A. S.: Auditory Scene Analysis: the Perceptual Organization of Sound. MIT Press (1990)
4. Chai, W.: Melody Retrieval on the Web. MSc Thesis, Massachusetts Institute of Technology (2001)
5. Ellis, D.: Prediction-Driven Computational Auditory Scene Analysis. PhD Thesis, Massachusetts Institute of Technology (1996)
6. Ghias, A., Logan, J., Chamberlin D., Smith, B. C.: Query by Humming: Musical Information Retrieval in an Audio Database. ACM Multimedia Conference (1995)
7. Goto, M.: A Predominant-F0 Estimation Method for CD Recordings: MAP Estimation Using EM Algorithm for Adaptive Tone Models. IEEE International Conference on Acoustics, Speech and Signal Processing (2001)
8. Handel, S.: Listening – An Introduction to the Perception of Auditory Events. MIT Press (1991)
9. Hartmann, W. M.: Signals, Sound and Sensation. AIP Press (1997)

10. Klapuri, A.: Multipitch Estimation and Sound Separation by the Spectral Smoothness Principle. IEEE International Conference on Acoustics, Speech and Signal Processing (2001)
11. Lyon, R. F.: A Computational Model of Filtering, Detection and Compression in the Cochlea. IEEE International Conference on Acoustics, Speech and Signal Processing (1982) 1282-1285
12. McAulay, R.J., Quatieri, T. F.: Speech Analysis/Synthesis based on a Sinusoidal Representation. IEEE Transactions on Acoustics, Speech and Signal Processing, Vol. 34(4) (1986) 744--754
13. Martin, K. D.: Automatic Transcription of Simple Polyphonic Music: Robust Front End Processing. 3rd Joint Meeting of the Acoustical Societies of America and Japan (1996)
14. Paiva, R. P., Mendes, T., Cardoso, A.: A Methodology for Detection of Melody in Polyphonic Musical Signals. 116th Audio Engineering Convention (2004)
15. Scheirer, E. D.: Music-Listening Systems. PhD Thesis, Massachusetts Institute of Technology (2000)
16. Serra, X.: Musical Sound Modeling with Sinusoids Plus Noise. In: Roads, C., Pope, S., Picialli, A., De Poli, G. (eds.): Musical Signal Processing (1997)
17. Slaney, M., Lyon, R. F.: On the Importance of Time – A Temporal Representation of Sound. In: Cooke, M., Beet, S., Crawford, M. (eds.): Visual Representations of Speech Signals (1993)
18. Slaney, M.: Auditory Toolbox: A Matlab Toolbox for Auditory Modeling Work (version 2). Technical Report, Interval Research Corporation (1998)
19. Smith, S.: The Scientist and Engineer's Guide to Digital Signal Processing. California Technical Publishing (1997)
20. Song, J., Bae, S. Y., Yoon, K.: Mid-Level Music Melody Representation of Polyphonic Audio for Query-by-Humming System. International Symposium on Music Information Retrieval (2002)
21. Uitdenbogerd, A. L., Zobel, J.: Music ranking techniques evaluated, Australasian Computer Science Conference (2002), 275-283

A New Probabilistic Spectral Pitch Estimator: Exact and MCMC-approximate Strategies

Harvey D. Thornburg and Randal J. Leistikow

Center for Computer Research in Music and Acoustics (CCRMA)
Department of Music
Stanford University
Stanford, CA, USA
{harv23, randal}@ccrma.stanford.edu

Abstract. We propose a robust probabilistic pitch (f_0) estimator in the presence of interference and low SNR conditions, without the computational requirements of optimal time-domain methods. Our analysis is driven by sinusoidal peaks extracted by a windowed STFT. Given f_0 and a reference amplitude (A_0), peak frequency/amplitude observations are modeled probabilistically in order to be robust to undetected harmonics, spurious peaks, skewed peak estimates, and inherent deviations from ideal or other assumed harmonic structure. Parameters f_0 and A_0 are estimated by maximizing the observations' likelihood (here A_0 is treated as a nuisance parameter). Some previous spectral pitch estimation methods, most notably the work of Goldstein [3], introduce a probabilistic framework with a corresponding maximum likelihood approach. However, our method significantly extends the latter in order to guarantee robustness under adverse conditions, facilitating possible extensions to the polyphonic context. For instance, our addressing of spurious as well as undetected peaks averts a sudden breakdown under low-SNR conditions. Furthermore, our assimilation of peak amplitudes facilitates the incorporation of timbral knowledge. Our method utilizes a hidden, discrete-valued descriptor variable identifying spurious/undetected peaks. The likelihood evaluation, requiring a computationally unwieldy summation over all descriptor states, is successfully approximated by a MCMC traversal chiefly amongst high-probability states. The MCMC traversal obtains virtually identical evaluations for the entire likelihood surface at a fraction of the computational cost.

1 Introduction

Fundamental frequency (f_0) estimation finds application in many areas of acoustics and digital audio. A robust estimation is especially relevant in music information retrieval tasks, where pitched signals may encounter significant interference from noise or other musical events.

To this end, we propose a high quality f_0 estimation in the presence of interference and poor signal-to-noise ratios without the computational load and storage costs of optimal time-domain methods. The input signal is preprocessed by a STFT followed by a sinusoidal peak extraction, outputting a list of frequency-amplitude pairs. The f_0 estimation operates directly on this peaklist.

Absent interference and irregularities, the k^{th} frequency in the observed peaklist will be kf_0. We denote this ideal, fictitious set of peaks the *input peaklist.* The actually observed peaklist, denoted the *output peaklist*, is perturbed by one or more of the following irregularities:

- **Undetected peaks:** A peak may be beneath the noise floor or masked due to interfering sinusoids close in frequency.
- **Spurious output peaks:** Interfering signals may themselves lead to additional peaks, or in a rare case, sidelobe peaks may be detected.
- **Skewed frequency estimates:** A natural inharmonicity may occur. Furthermore, interfering sinusoids may perturb STFT frequency peak estimates even when a peak is detected. An additional bias may incur due to the parabolic interpolation between STFT bins.

Given a probabilistic model of the output peaklist conditional upon f_0 and A_0, we estimate f_0 via maximum likelihood, treating A_0 as a nuisance parameter.

Some pioneering work in spectral pitch estimation, for instance Goldstein's method [3], has introduced a probabilistic framework operating on STFT peaks. While the latter approach does account for the possibility of undetected peaks, the issue of spurious peaks so far remains unaddressed. As we demonstrate, a proper accounting for spurious peaks becomes vitally important under low-SNR conditions, where approaches which do not consider them often experience a sudden breakdown. A practical context arises, for instance, when detecting primary pitch formations in a polyphonic setting. A further advantage of our approach is the ability to utilize peak amplitudes via prior information concerning timbre. Appendix 5 provides a detailed comparison between our method and a suitable generalization of Goldstein's method under identical computational requirements.

2 Probabilistic Model for the Output Peaklist

Figure 1 illustrates our probabilistic model for the output peaklist given $f_0 \in [0, \pi]$, $A_0 \in \mathbb{R}^+$.

2.1 The Hidden Descriptor

We introduce a hidden *descriptor*

$$D \in \mathcal{D} = \cup_{N=0}^{\infty} \{\text{'I'}, \text{'B'}, \text{'O'}\}^N \tag{1}$$

specifying linkage between the input and output peaklists, as in Figure 2.

Let index j refer to the increasing-frequency sorting order of the aggregation of all input peaks and spurious output peaks. Then $D(j) =$ 'I' labels an undetected input peak; $D(j) =$ 'B' labels an output peak linked to some input peak, and $D(j) =$ 'O' labels a spurious output peak.

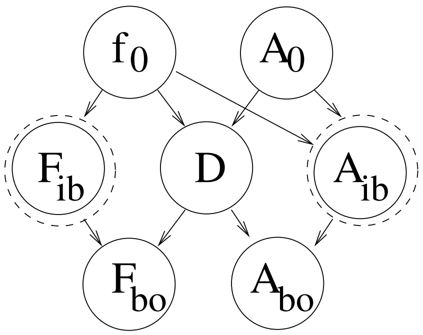


Fig. 1. *Probabilistic model: descriptor, input and output peaks. Dotted circles describe degenerate (deterministic) relations; all other relations are probabilistic.*

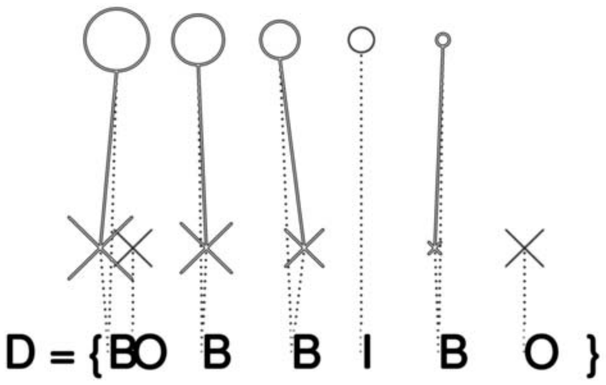


Fig. 2. *Linked input and output peaks with descriptor. Sizes of circles and X's represent relative peak amplitudes.*

Let F_{bo} and A_{bo} be parallel vectors of output peak frequency and amplitude observations. We denote their observation spaces:

$$F_{bo} \in \mathcal{F} = \cup_{N_{bo}=1}^{\infty} [0, \pi]^{N_{bo}} \tag{2}$$

$$A_{bo} \in \mathcal{A} = \cup_{N_{bo}=1}^{\infty} \{\mathbb{R}^{+}\}^{N_{bo}} \tag{3}$$

Similarly, let F_{ib} and A_{ib} be the (fictitious) input peak lists. F_{ib} is generated according to a harmonic template given f_0:

$$F_{ib}(j_{ib}) = j_{ib} \cdot f_0, \quad j_{ib} \in \{1, 2, \dots \lfloor \pi/f_0 \rfloor\} \tag{4}$$

A_{ib} consists of a parallel collection of amplitudes, dependent on a reference amplitude A_0 and decay parameter c_A:

$$A_{ib}(j_{ib}) = A_0 c_A^{j_{ib}-1} \tag{5}$$

Now, we define mapping indices

$$j_{ib}(j) \overset{\Delta}{=} \sum_{l=1}^{j} 1_{\{D(l)=\text{'B'}\}\cup\{D(l)=\text{'I'}\}}$$

$$j_{bo}(j) \overset{\Delta}{=} \sum_{l=1}^{j} 1_{\{D(l)=\text{'B'}\}\cup\{D(l)=\text{'O'}\}}$$

$$j_{b}(j) \overset{\Delta}{=} \sum_{l=1}^{j} 1_{\{D(l)=\text{'B'}\}} \tag{6}$$

Then if $D(j) = \text{'B'}$, the j_{bo}^{th} output peak, i.e., the observation $\{F_{bo}(j_{bo}(j)), A_{bo}(j_{bo}(j))\}$ is linked to the j_{ib}^{th} input peak, i.e., that with frequency $j_{ib}(j)f_0$. Otherwise the output peak is considered spurious.

Knowledge of D enables computation of the conditional likelihoods $P(F_{bo}, A_{bo}|D, f_0, A_0)$ and $P(D|f_0, A_0)$. Our goal is to choose f_0 and A_0 such as to maximize the *unconditional* likelihood, $P(F_{bo}, A_{bo}|f_0, A_0)$, which entails a summation over $\mathcal{D}$:

$$P(F_{bo}, A_{bo}|f_0, A_0) = \sum_{D\in\mathcal{D}} P(D|f_0, A_0)P(F_{bo}, A_{bo}|D, f_0, A_0) \tag{7}$$

Before addressing the summation, we shall make explicit the forms of $P(F_{bo}, A_{bo}|D, f_0, A_0)$ and $P(D|f_0, A_0)$.

2.2 Specification of $\mathbf{P(F_{bo}, A_{bo}|D, f_0, A_0)}$

If F_{bo} and A_{bo} are *invalid* with respect to D and f_0, then $P(F_{bo}, A_{bo}|D, f_0, A_0) \overset{\Delta}{=} 0$. The relevant validity conditions are as follows:

- **V1** The number of output peaks (length of F_{bo}) must equal the number of 'B' plus 'O' symbols in D, i.e., $N_{bo}(D)$.
- **V2** The frequency sorting order of the aggregation of all input and spurious output peaks in $\{F_{ib}, F_{bo}\}$ must match the order of corresponding 'I' and 'O' symbols in D.

Otherwise, for valid F_{bo}, A_{bo}, our model admits the following factorization:

$$P(F_{bo}, A_{bo}|D, f_0, A_0) = P(F_{bo}|D, f_0)P(A_{bo}|D, A_0) \tag{8}$$

In (8), frequency and amplitude deviations of output peaks are modeled as conditionally independent given D, f_0 and A_0. Furthermore, we assume sufficient spacing in frequency to prevent significant interactions between output peaks; hence $P(F_{bo}|D, f_0)$ and $P(A_{bo}|D, A_0)$ factor as product distributions over the individual peak observations:

$$P(F_{bo}|D, f_0) = \prod_{j_{bo}=1}^{N_{bo}} P(F_{bo}(j_{bo})|D, f_0) \tag{9}$$

$$P(A_{bo}|D, A_0) = \prod_{j_{bo}=1}^{N_{bo}} P(A_{bo}(j_{bo})|D, A_0) \tag{10}$$

In case of a linked output peak, i.e., $D(j) =$ 'B', $F_{bo}(j_{bo}(j))$ is modeled as a Gaussian dependence on the frequency of the corresponding input peak:

$$F_{bo}\left(j_{bo}(j)|D(j) = \text{'B'}\right) \sim \mathcal{N}\left(F_{ib}(j_{ib}(j)), \sigma^2_{f,b}\right) \tag{11}$$

If the output peak is spurious, i.e., $D(j) =$ 'O', the frequency is modeled as uniform $[0, \pi]$, via maximum entropy argument:

$$F_{bo}\left(j_{bo}(j)|D(j) = \text{'O'}\right) \sim U(0, \pi) \tag{12}$$

The output peak amplitude for a linked output peak is modeled according to the result expected from a sinusoid plus Gaussian noise in the time domain. In the frequency domain, $2A^2_{bo}(j_{bo}(j))/\sigma^2_{A,b}$ follows the distribution of a noncentral χ^2_2 with noncentrality parameter $2A^2_{ib}(j_{ib}(j))/\sigma^2_{A,b}$ [4].

$$P\left(\left.\frac{2A^2_{bo}\left(j_{bo}(j)\right)}{\sigma^2_{A,b}}\right| D(j) = \text{'B'}\right) \sim \chi^2_{2,\frac{2A^2_{ib}(j_{ib}(j))}{\sigma^2_{A,b}}} \tag{13}$$

Recall that, via (5), $A_{ib}(j)$ depends on the nuisance c_A. A preferable solution is to marginalize c_A with respect to some noninformative prior; however we have not yet taken this step. Instead, we obtain acceptable results by fixing $c_A \approx 0.95$ and using $\sigma^2_{A,b} = 0.25$.

The amplitude of a spurious output peak is modeled as Gaussian with variance $\sigma^2_{A,o}$. In this case, $2A^2_{bo}(j_{bo}(j))/\sigma^2_{A,o}$ follows a *central* χ^2_2:

$$P\left(\left.\frac{2A^2_{bo}\left(j_{bo}(j)\right)}{\sigma^2_{A,o}}\right| D(j) = \text{'O'}\right) \sim \chi^2_{2,0} \tag{14}$$

2.3 Specification of $\mathbf{P(D|f_0, A_0)}$

If D is invalid with respect to f_0, we define $P(D|f_0, A_0) = 0$. The corresponding validity condition is:

– **V3** The number of 'I' plus 'B' symbols in D, i.e., $N_{ib}(D)$, must equal the number of input peaks. The latter, fixed according to Nyquist considerations (4), eliminates redundancies, as a missing input peak is equivalent to an unlinked peak.

Otherwise, define D_{ib} as D with the 'O' symbols removed, i.e., for all j such that $D(j) =$ 'I'or'B', $D_{ib}(j_{ib}(j)) = D(j)$. The factorization $P(D|f_0, A_0) = P(D_{ib}|f_0, A_0)P(D|D_{ib}, f_0, A_0)$ separates the input peaks' survival $P(D_{ib}|f_0, A_0)$ from the spurious output peaks' generation $P(D|D_{ib}, f_0, A_0)$. The survival distribution is modeled as independent Bernoulli with decaying survival probability:

$$P(D_{ib}(j_{ib}) = \text{'B'}|f_0, A_0) = \phi_b^{j_{ib}} \tag{15}$$

Now, let $N_o(D)$ be the number of 'O' symbols in D. Since spurious frequencies are generated according to a Poisson process,

$$P(N_o(D)|D_{ib}, f_0, A_0) = e^{-\lambda_o} \frac{\lambda_o^{N_o(D)}}{N_o(D)!} \tag{16}$$

where λ_o is the mean number of spurious output peaks on $[0, \pi]$. As

$$P(D|D_{ib}, f_0, A_0) = P(D|N_o(D), D_{ib}, f_0, A_0)P(N_o(D)|D_{ib}, f_0, A_0) \tag{17}$$

it remains to specify the latter part of this factorization. The latter depends only on the number of 'O' symbols interleaved between each 'I' or 'B' symbol, i.e., on $\{N_{o,k}(D)\}_{k=0}^{N_{ib}(D)}$ where $N_{o,k}(D) = \sum_{\{j:j_{ib}(j)=k\}} \mathbf{1}_{\{D(j)=\text{'O'}\}}$. Since, conditional upon $N_o(D)$, the spurious frequency locations are iid uniform on $[0, \pi]$, it follows that

$$P(D|N_o(D), D_{ib}, f_0, A_0) {=} P(\{N_{o,k}(D)\}_{k=0}^{N_{ib}(D)} |D_{ib}, f_0, A_0) \tag{18}$$

admits a multinomial distribution:

$$P(\{N_{o,k}(D)\}_{k=0}^{N_{ib}(D)} |D_{ib}, f_0, A_0) = \frac{N_o(D)! \prod_{k=0}^{N_{ib}(D)} \rho_k^{N_{o,k}(D)}}{\prod_{k=0}^{N_{ib}(D)} N_{o,k}(D)!} \tag{19}$$

where ρ_k, the probability of a single 'O' symbol being interleaved in the k^{th} interval delimited by an 'I' or 'B' symbol, is given by:

$$\rho_k = \begin{cases} F_{ib}(1)/\pi, & k = 0 \\ 1 - F_{ib}(N_{ib}(D))/\pi, & k = N_{ib}(D) \\ \left[F_{ib}(k+1) - F_{ib}(k)\right]/\pi, & \text{otherwise} \end{cases} \tag{20}$$

As a result,

$$P(D|f_0, A_0) = e^{-\lambda_o}\lambda_o^{N_o(D)} \prod_{k=1}^{N_{ib}(D)} \frac{\left(\phi_b^k\right)^{\mathbf{1}_{\{D_{ib}(k)=\text{'B'}\}}} \left(1 - \phi_b^k\right)^{\mathbf{1}_{\{D_{ib}(k)=\text{'I'}\}}}}{\rho_k^{-N_{o,k}(D)} N_{o,k}(D)!} \tag{21}$$

2.4 Implication of Validity Conditions

If any one of **V1-V3** fail, $P(F_{bo}, A_{bo}, D|f_0, A_0) = 0$. Thus, D can be skipped in the summation (7). Otherwise a set of D satisfying all validity conditions is isomorphic to the set of nonintersecting linkmaps between F_{ib} and F_{bo}, the latter to be described. While the descriptor provides a convenient mechanism for the evaluation of the conditional likelihoods which form the unconditional likelihood, via (7), the linkmap representation enables a straightforward enumeration of valid descriptors.

Let $N_b(D)$ be the number of 'B's in D. The *linkmap*, denoted $L \in \mathbb{Z}^{+[2\times N_b(D)]}$, is a matrix consisting of an input row, $L(i,:)$, and an output row, $L(o,:)$. Each column describes a linkage between an input and an output peak. A linkage must occur for each j for which $D(j) =$ 'B'. We set $L(i, j_b(j)) = j_{ib}(j)$ and $L(o, j_b(j)) = j_{bo}(j)$.

Though a unique linkmap arises from a given descriptor, without **V1-V3**, a given linkmap may arise from more than one descriptor.

For example, let

$$F_{ib} = \{0.5, 1, 1.5, 2, 2.5\}$$
$$F_{bo} = \{0.7, 1, 1.3, 2.8\}$$

and suppose $L(i,:) = [2, 4]$; $L(o,:) = [2, 4]$. This example is diagrammed in Figure 3.

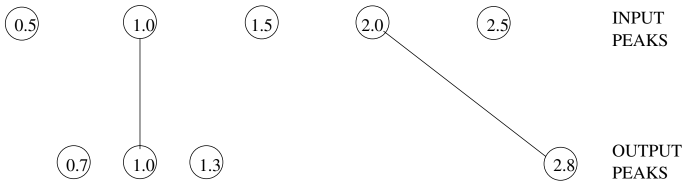


Fig. 3. *Example linkmap*

Any of the following descriptors may yield L:

$$D_1 = \{\text{'I','O','B','O','I','B','I'}\}$$
$$D_2 = \{\text{'I','O','B','I','O','B','I'}\}$$
$$D_3 = \{\text{'I','O','B','O','I','B','I', 'I', 'O','I'}\}$$

However, only D_1 satisfies **V1-V3**.

The unique specification of D follows:

- $N_b(D)$ is fixed by L;
- $N_{ib}(D)$ is fixed by f_0 via **V3**;
- $N_{bo}(D)$ is fixed by F_{bo} via **V1**;
- the relative ('I' $\leftrightarrow$ 'B') and ('B' $\leftrightarrow$ 'O') orderings are fixed by L;
- the ('I' $\leftrightarrow$ 'O') ordering is fixed by f_0 and F_{bo} via **V2**.

Since the number of 'I','B',and 'O' symbols are fixed by L, as well as all pairwise orderings, the D sequence itself is fixed. Thus, **V1 - V3** establish the isomorphism $D \leftrightarrow L$.

3 Enumeration Strategies

To evaluate the unconditional likelihood via (7), we must enumerate all valid descriptors, i.e., those for which $P(D|f_0, A_0) > 0$ and $P(F_{bo}, A_{bo}|D, f_0, A_0) > 0$. We consider an exact evaluation of (7) via brute-force enumeration, then via an approximate MCMC scheme. The latter constructs a random walk amongst the descriptors which contribute most of the probability to this summation.

3.1 Exact (Brute Force) Enumeration

Due to the linkmap $\leftrightarrow$ valid descriptor isomorphism, a compact and exhaustive enumeration of descriptors is achieved simply by enumerating all possible linkmaps between N_{ib} input peaks and N_{bo} output peaks. Let

$$m \in \{0, 1, ...\min(N_{ib}, N_{bo})\}$$

be the outermost enumeration index representing the number of links. For each m, we enumerate the $\binom{N_{ib}}{m}$ sets of m from N_{ib} input peaks in an outer loop and the $\binom{N_{bo}}{m}$ sets of m from N_{bo} output peaks in an inner loop. The extraction of the descriptor corresponding to a given linkmap is straightforward. The total number of linkmaps, hence valid descriptors, is as follows.

$$\#\{D\} = \sum_{m=0}^{\min(N_{ib},N_{bo})} \binom{N_{ib}}{m}\binom{N_{bo}}{m} \tag{22}$$

For small N_{ib} and N_{bo}, (e.g., $N_{ib}, N_{bo} \leq 7$), the enumeration is not prohibitive. Exact enumeration proves useful in comparing approximate strategies. Unfortunately, the nature of the combinatoric explosion becomes evident when N_{ib} and N_{bo} simultaneously become large. For instance, let $N_{ib} = N_{bo} = N$. The summation in (22) simplifies accordingly:

$$\#\{D\} = \sum_{m=0}^{N} \binom{N}{m}^2 = \binom{2N}{N} \tag{23}$$

As discussed by Knuth, et al. [5], Stirling's approximation reveals the following asymptotic behavior:

$$\binom{2N}{N} = \frac{4^N}{\sqrt{\pi N}}\left[1 - \mathcal{O}\left(\frac{1}{N}\right)\right] \tag{24}$$

Hence, the number of required descriptor enumerations grows exponentially with a common number of input/output peaks, to first order in the exponent.

3.2 Results from Exact Enumeration

An $A4$ piano tone, allegedly at 440 Hz, is recorded at 44.1 kHz with -14 dB white noise added. A 227 ms frame is used to derive a pitch estimate. We set $A_0 = A_1/c_A$, A_1 being the highest (fundamental) peak amplitude. Additionally, we specify:

- $\phi_b = 0.55$
- $\lambda_o = 10.0$
- $c_A = 0.35$
- $\sigma^2_{f,b} = (0.05 f_0)^2$
- $\sigma^2_{A,b} = (0.5 A_0)^2$
- $\sigma^2_{A,o} = (0.05 A_0)^2$

The analysis is artificially truncated to the first seven input and output peaks, to facilitate a tractable computation. As a result, each candidate f_0 requires exactly 3432 likelihood computations. It should be noted that two of the seven output peaks appear spurious.

The solid curve in Figure 4 displays the 0.05 power of the total likelihood vs. f_0, while the dotted curves display likelihood contributions summed over the top $1-3$ descriptors. (The 0.05 power is taken to improve readability of local maxima.) Thanks to the scaling, the dotted curves are indistinguishable from the main curve. The maximum likelihood estimate is $f_0 = 0.0628$ rad/sample (441 Hz), while other local maxima indicate subharmonics.

To assess the contributions from the top three descriptors, Figure 5 shows their fraction of total likelihood vs. f_0. Table 1 displays also the total likelihood fraction averaged over all f_0, as well as the fraction of f_0 for which at least 99% of the total likelihood is contributed by the reduced set of descriptors.

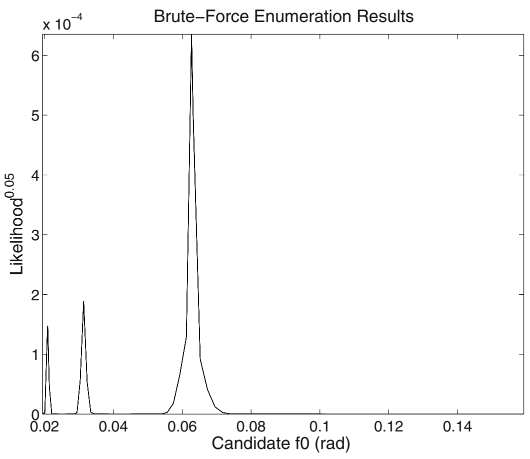


Fig. 4. *Likelihood surface from exact enumeration: 7 peaks, 2 spurious*

Since indeed most of the likelihood concentrates in a few descriptors, computations may be reduced by an approximate MCMC navigation scheme traversing chiefly among these descriptors. The latter is detailed in the following section.

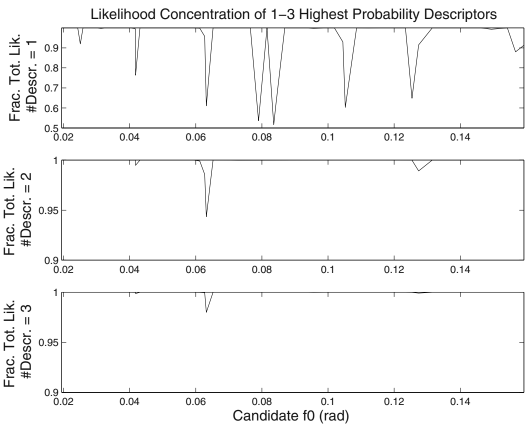


Fig. 5. *Likelihood concentration for 1-3 descriptors*

#Descriptors	Average Likelihood Fraction	Exceed 99% Fraction
1	.9581	.8312
2	.9970	.9610
3	.9997	.9870

Table 1. *Likelihood concentration of 1-3 highest probability descriptors*

4 MCMC Approximate Enumeration

In any MCMC scheme, the purpose is to construct a Markovian transition rule, $P(D_{k+1}|D_k)$ which preserves a desired stationary distribution [2]. We denote this distribution as $\pi(D)$. We desire that $\pi(D)$ concentrate in descriptors yielding a relatively high joint likelihood. To this end, we specify:

$$\pi(D) \propto \left[P(D|f_0, A_0)P(F_{bo}, A_{bo}|D, f_0, A_0)\right]^K \tag{25}$$

where $K > 1$. K must not be too large, however, since $K \to \infty$ implies $\pi(D)$ concentrates on the set of descriptors yielding *only* the maximum joint likelihood, which as Table 1 shows, can occasionally fail to capture most of the likelihood.

If $P(D_{k+1}|D_k)$ is irreducible, the resultant Markov chain will converge in distribution to $\pi(D)$ [1]. To facilitate fast convergence, D_0 is initialized by the McAulay-Quatieri (MQ) approximate matching algorithm [6].

$P(D_{k+1}|D_k)$ follows the Metropolis-Hastings rule [2]. A candidate D'_k is sampled $\sim q(D'_k|D_k)$, then D'_k is accepted (i.e., $D_{k+1} = D'_k$) with probability $\min(1, r(D_k, D'_k))$; otherwise $D_{k+1} = D_k$, where:

$$r(D_k, D'_k) = \frac{\pi(D'_k)q(D_k|D'_k)}{\pi(D_k)q(D'_k|D_k)} \tag{26}$$

The sampling distribution, $q(D'_k|D_k)$, operates directly on the set of linkmaps $\{L\}$, which exist in one-to-one correspondence with valid D. Let $L_k \leftrightarrow D_k$ and

$L'_k \leftrightarrow D'_k$. Then $q(D'_k|D_k)$ is chosen to satisfy some notion of adjacency on $\{L_k, L'_k\}$. Any of the following moves may generate L'_k:

1. **Remove a link**: Delete a column in L_k.
2. **Add a nonintersecting link**: If possible, choose any unlinked pair for which a possible linkage does not intersect the current set of links, and insert this pair as a column in L_k. To simplify the argument, consider L_k as augmented with the set of "phantom" boundary links, i.e., $L_k(i,0) = L_k(j,0) = 0$; $L_k(i, N_b(D_k)+1) = N_{ib}(D_k)$; $L_k(o, N_b(D_k)+1) = N_{bo}(D_k)$. The nonintersection condition is satisfied if and only if there exists $m \in \{0,\ldots,N_b(D_k)\}$, $j_{ib} \in \{1,\ldots,N_{ib}(D_k)\}$, and $j_{ob} \in \{1,\ldots,N_{ob}(D_k)\}$ for which $L_k(i,m) < j_{ib} < L_k(i,m+1)$ and $L_k(o,m) < j_{ob} < L_k(o,m+1)$.
3. **Switch a link to adjacent input** If there exists $m \in \{1,\ldots,N_b(D_k)\}$ for which $L_k(i,m-1) < L_k(i,m) - 1$, decrement $L_k(i,m)$. Similarly, if $L_k(i,m)+1 < L_k(i,m+1)$, increment $L_k(i,m)$.
4. **Switch a link to adjacent output** The process is identical to (3), except the switch position is $L_k(o,m)$.

Figure 6 illustrates one example from each of the aforementioned categories of move possibilities, where L_k originates from the linkmap example of Figure 3.

Given the current linkmap, $N_{ib}(D_k)$, and $N_{bo}(D_k)$, each set of move possibilities is computed for each category. A category is selected equiprobably over the categories with at least one move possibility, then a move is selected equiprobably amongst the possibilities for that category.

If any move can occur with positive probability, and each of the possible linkmaps generates a *strictly valid* D'_k, i.e., $\pi(D'_k) > 0$, the irreducibility of the chain is guaranteed. Irreducibility follows via $L \leftrightarrow$ valid D isomorphism and the fact we may reach any linkmap from any other linkmap by removing then adding links one by one with nonzero probability.

4.1 Results from MCMC Approximate Enumeration

Under identical conditions generating Figures 4-5, Figures 7-8 respectively compare the MCMC vs. pure MQ initialization and exact enumeration likelihood functions, and display the MCMC and MQ likelihoods as a fraction of total likelihood. We seem to obtain faster convergence upon varying K according to the "annealing" schedule:

$$K(0) = 0.05$$
$$K(k) = \min(1.03K(k-1), 5.0)$$

The visible dotted line in Figure 7 represents the MQ likelihood. Note that the MQ likelihood seems close to the exact likelihood about the true f_0 and subharmonics; however, as shown in Figure 8, almost none of the likelihood is captured for other f_0. The MCMC likelihood is visually indistinguishable from

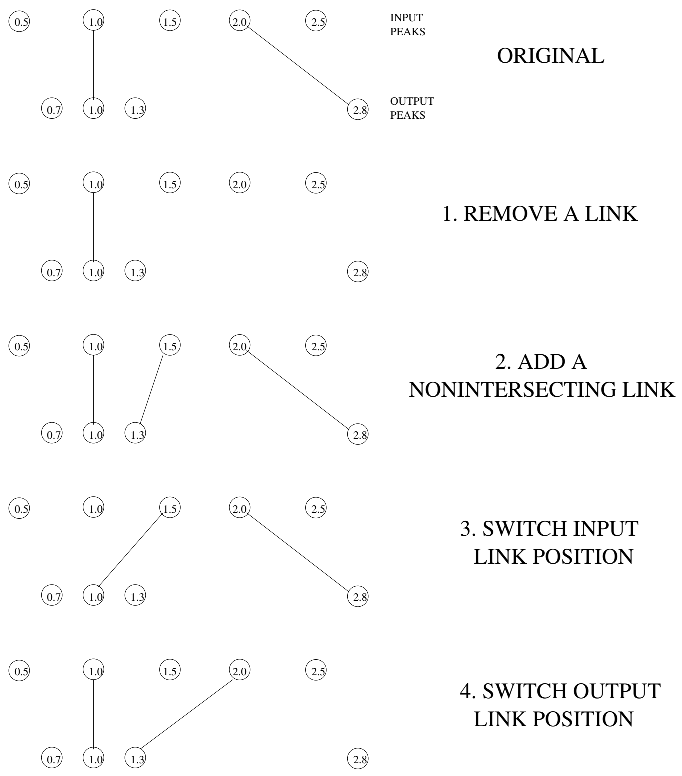

0.5
1.0
1.5
2.0
2.5
INPUT PEAKS
ORIGINAL
0.7
1.0
1.3
2.8
OUTPUT PEAKS
1. REMOVE A LINK
2. ADD A NONINTERSECTING LINK
3. SWITCH INPUT LINK POSITION
4. SWITCH OUTPUT LINK POSITION

Fig. 6. *Categories of move possibilities*

the exact result. Likelihood concentration results are summarized in Table 2. The MCMC obtains significant computational savings at essentially no change in the f_0-likelihood function: hashing likelihood computations for previously visited descriptors obtains an average of 22.38 likelihood computations per f_0, vs. exactly 3432 computations per f_0 for the brute force.

Method	Average Likelihood Fraction	Exceed 99% Fraction
MQ Init. only	.1994	.1948
MCMC	$1 - 3.1819 \cdot 10^{-13}$	1

Table 2. *Likelihood concentration of MCMC and MQ initialization*

5 Conclusions and Subsequent Research

We have developed a probabilistic spectral pitch estimator robust to both undetected harmonics and spurious peaks. Our method computes not only a pitch

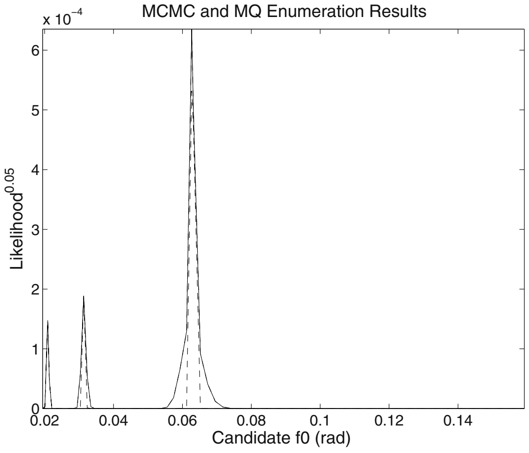


Fig. 7. *Likelihood surfaces: MCMC and MQ vs. exact enumeration: 7 peaks, 2 spurious*

estimate, but evaluates an entire likelihood surface. In other words, the unconditional likelihood of the STFT peak observations, $P\left(F_{bo}, A_{bo}|f_0\right)$, may be evaluated for *any* candidate f_0 (given that the nuisance A_0 has been eliminated).

Besides facilitating pitch estimation, our likelihood evaluation may comprise a vital structural element in a hierarchical Bayesian inference for note values. The latter proves quite useful when we wish to integrate information across STFT frames, or exploit additional knowledge from musical structure. In figure 9, we illustrate a hypothetical Bayesian extension to melody tracking.

Here, $M^{(t)}$ represents a labeled semitone value (e.g., $A4$, $C5$, etc.) responsible for generating the t^{th} STFT frame: $\left(F_{bo}^{(t)}, A_{bo}^{(t)}\right)$. Our goal is to compute the posterior of $M^{(t)}$ given all STFT frames: $P\left(M^{(t)}\middle|F_{bo}^{(1:T)}, A_{bo}^{(1:T)}\right)$. By so doing, a framewise maximization of this posterior (i.e., for each $M^{(t)}$) "estimates" a melody $\hat{M}^{(1:T)}$ which minimizes the expected number of note errors.

The posterior inference $P\left(M^{(t)}\middle|F_{bo}^{(1:T)}, A_{bo}^{(1:T)}\right)$ is facilitated by the hidden Markovian factorization of the joint as indicated in Figure 9:

$$P\left(M^{(1:T)}, F_{bo}^{(1:T)}, A_{bo}^{(1:T)}\right) = P\left(M^{(1)}\right) \times \prod_{t=2}^{T} P\left(M^{(t)}\middle| M^{(t-1)}\right) \times \prod_{t=1}^{T} P\left(F_{bo}^{(t)}, A_{bo}^{(t)}\middle| M^{(t)}\right) \qquad (27)$$

Given the *prior* $P\left(M^{(1)}\right)$, the *interframe dependences* $P\left(M^{(t)}\middle| M^{(t-1)}\right)$, and the STFT *frame likelihoods* $P\left(F_{bo}^{(t)}, A_{bo}^{(t)}\middle| M^{(t)}\right)$, one may apply standard hidden

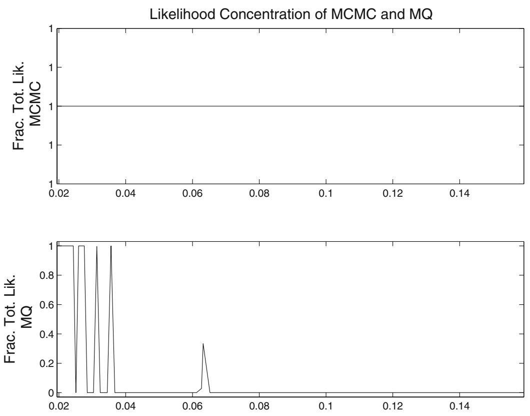


Fig. 8. *Fraction of total likelihood covered by MCMC traversal*

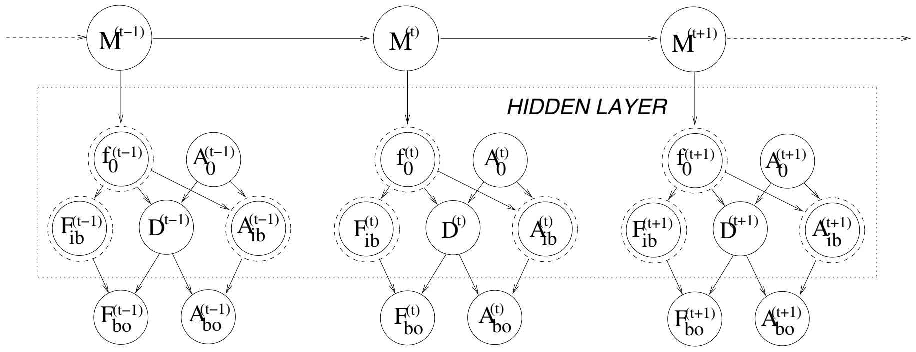


Fig. 9. *Proposed melody tracker*

Markov model (HMM) inference strategies, for instance the forward-backward algorithm [7], to compute the desired posterior.

The structure of interframe dependences factors across two levels: literal frame-to-frame dependences, and dependences across note transitions. Since the STFT frames are usually quite short, many frames may appear between note transitions. Hence, one may exploit significant additional structure by concentrating $P\left(M^{(t)} \middle| M^{(t-1)}\right)$ about the possibility $M^{(t)} = M^{(t-1)}$, while maintaining $P\left(M^{(t)} \neq M^{(t-1)} \middle| M^{(t-1)}\right)$ at a small but nonzero value to allow for occasional note transitions. At the level of note transitions, the distribution $P\left(M^{(t)} \middle| M^{(t-1)}\right)$ *given* that a transition has occurred, i.e.: $M^{(t)} \neq M^{(t-1)}$, may be further restricted by specific knowledge or rules concerning melodic structure.

Ultimately, our ability to compute the posterior $P\left(M^{(t)} \middle| F_{bo}^{(1:T)}, A_{bo}^{(1:T)}\right)$ depends on our ability to evaluate frame likelihoods. Since the relation between note number and fundamental frequency is deterministic,

$$P\left(F_{bo}^{(t)}, A_{bo}^{(t)}\middle| M^{(t)}\right) = P\left(F_{bo}^{(t)}, A_{bo}^{(t)}\middle| f_0\right) \tag{28}$$

either our exact enumeration or MCMC traversal strategy may be used to evaluate the frame likelihood $P\left(F_{bo}^{(t)}, A_{bo}^{(t)}\middle| M^{(t)}\right)$.

Such melody tracking extensions are presently under investigation. A further extension is to generalize the frame likelihood evaluation to the multipitch case. The latter may be embedded in a Bayesian chord recognition engine, analogously to the manner in which our single-pitch evaluation is embedded in the proposed melody tracker. Preliminary chord recognition results, presented in [8], seem quite promising.

References

1. Cinlar, E. *Introduction to Stochastic Processes*, Prentice-Hall, Englewood Cliffs, NJ, 1975.
2. Fitzgerald, W.J. "Markov chain Monte Carlo methods with applications to signal processing", Elsevier Signal Processing 81(1):3-18, Jan. 2001.
3. Goldstein, J. "An optimum processor theory for the central formation of the pitch of complex tones", J. Acoust. Soc. Amer. 54:1496-1516, 1973.
4. Hory, C., Martin, N., and Chehikian, A. "Spectrogram segmentation by means of statistical features for non-stationary signal interpretation", IEEE Trans. ASSP 50(12):2915-2925, Dec. 2002.
5. Knuth, D., Vardi, I., and Richberg, R. "6581 (The asymptotic expansion of the middle binomial coefficient)", American Mathematical Monthly, 97(7):626-630, Aug/Sep. 1990
6. McAulay, R.J. and Quatieri, T.F. "Speech analysis/synthesis based on a sinusoidal representation", IEEE Trans. ASSP 34(4):744-754, Aug. 1986.
7. Rabiner, L.R. "A tutorial on hidden Markov models and selected applications in speech recognition", Proceedings of the IEEE, 77(2):257-286, 1989.
8. Leistikow, R., Thornburg, H., et al. "Bayesian Identification of Closely-Spaced Chords from Single-Frame STFT Peaks", Proc. 7th International Conference on Digital Audio Effects (DAFx'04), Naples, 2004.

Appendix: Comparison with Goldstein's Method

The probabilistic framework introduced by Goldstein [3] may be recast into a framework similar to ours, to facilitate comparisons. Let f_0, as usual, denote the fundamental frequency. The output peaklist, denoted F_o, consists of a list of observed peak frequencies, sorted by increasing frequency:

$$F_o = \left\{f_{(o,1)}, \ldots, f_{(o,N_o)}\right\} \tag{29}$$

Here $f_{o,k}$ denotes the frequency of the k^{th} observed peak and N_o is the number of observed peaks. Let F_i, the (fictitious) template or input peaklist, be given accordingly:

$$F_i = \left\{f_{(i,1)}, \ldots, f_{(i,N_i)}\right\} \tag{30}$$

where

$$f_{(i,k)} = kf_0 \tag{31}$$

and N_i is the number of input peak candidates considered.

A descriptor is necessary to express linkage between input and output peaks. Define

$$D = \{n_1, \ldots, n_{N_o}\} \tag{32}$$

where each $n_k \in \mathbb{Z}^+$ signifies that the k^{th} output peak corresponds to the n_k^{th} harmonic of f_0. The idea is each output peak frequency, $f_{(o,k)}$, is a noisy observation of the input peak with frequency $f_{(i,n_k)} = n_k f_0$.

The probabilistic dependence structure is represented in Figure 10.

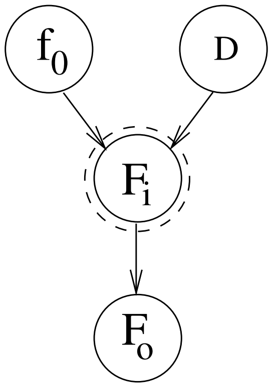


Fig. 10. *Equivalent probabilistic model: Goldstein's method*

As the observation $f_{(o,k)}$ depends only on n_k and f_0, given D, $P(F_o|F_i)$ factors into a product distribution over individual Gaussians for each peak:

$$P(F_o|F_i, D) = \prod_{k=1}^{N_o} \mathcal{N}\left(f_{(i,n_k)},\ \sigma_k^2\right) \tag{33}$$

Since $f_{(i,n_k)}$ is deterministically generated by D and f_0, (33) simplifies as follows.

$$P(F_o|f_0, D) = \prod_{k=1}^{N_o} \mathcal{N}\left(n_k f_0,\ \sigma_k^2\right) \tag{34}$$

Noise variances $\{\sigma_k^2\}_{k=1}^{N_o}$ are unknown and hence become nuisance parameters for the estimation. In [3], the issue is resolved by constraining each variance to be proportional to the corresponding input peak frequency:

$$\sigma_k^2 = Kn_k f_0 \tag{35}$$

The rationale is that due to inharmonicity structure, more uncertainty is generated in the frequencies of high-number harmonics relative to the frequencies of low-number harmonics. Hence, (34) simplifies accordingly, with K the only remaining nuisance parameter:

$$P(F_o|f_0, K, D) = \prod_{k=1}^{N_o} \mathcal{N}\left(n_k f_0,\ K n_k f_0\right) \tag{36}$$

The tolerance for uncertain inharmonicity structure provided by Goldstein's approach remains so far unaddressed by our approach. However, it seems quite valid, and we plan to correct our distributional specification for linked output peak frequencies along the lines of (36) in a subsequent revision.

Thanks to (36), with D fixed, it can be shown that the likelihood maximization with respect to f_0 is actually invariant to K. As a result, $\hat{f}_{0,D}$, the maximum likelihood estimate of f_0 with D fixed, obtains in closed form.

$$\hat{f}_{0,D} = \frac{\sum_{k=1}^{N_o} \left(f_{(o,k)}/\hat{n}_k\right)^2}{\sum_{k=1}^{N_o} f_{(o,k)}/\hat{n}_k} \tag{37}$$

In Goldstein's approach [3], D, as in our approach, is treated as a nuisance parameter. While we marginalize D with respect to some prior, $P(D|f_0)$, Goldstein includes D jointly in the likelihood maximization. Since we can maximize in any order, the unconditional estimate of f_0 obtains:

$$\hat{f}_0 = \max_D \hat{f}_{0,D} \tag{38}$$

The existence of a closed form solution for $\hat{f}_0$ removes the need to construct a search grid over candidate values for f_0, thus drastically reducing the number of computations necessary for the raw pitch estimate. However, the unconditional likelihood evaluation is itself often of interest, either to express pitch disambiguation, or to facilitate the Bayesian melody tracking briefly proposed in Section 5.

The maximization over D (38) induces a combinatoric explosion paralleling that of our exact enumeration. In particular, the choice $N_i = 2N_o$ requires an identical number of descriptor candidates to be enumerated. Since each D effectively represents a choice of N_o objects from N_i, the choices $N_o = N$ and $N_i = 2N$ imply

$$\#\left\{D\right\} = \binom{2N}{N} \tag{39}$$

We note that (39) matches the number of descriptor candidates required by our exact enumeration (23).

5.1 Results Comparison: Exact Enumeration Vs. Goldstein's Method

Goldstein's method determines jointly the most likely f_0 and D. To compare against our exact enumeration in terms of qualitative aspects such as pitch disambiguation, we must be able to evaluate the likelihood of the output peaks for any f_0. Therefore, we develop a suitable generalization of Goldstein's method enabling the unconditional likelihood for *any* f_0 to be evaluated in a manner consistent with the evaluation for the *most likely* f_0. In the latter case, the conditional likelihood given D, K is maximized with respect to D and invariant to K. Since the invariant condition may fail for a general f_0, we obtain the unconditional likelihood for a general f_0 by maximizing over D *and* K:

$$P(F_o|f_0) \overset{\Delta}{=} \max_{K,D} \ P(F_o|f_0, K, D) \tag{40}$$

The generalization (40) allows us to compare the unconditional likelihood surfaces produced by Goldstein's method against those generated by our exact enumeration. By so doing, we may ascertain robustness under various interference conditions. All examples utilize a 227 ms frame consisting of a 440 Hz piano tone with −14 dB white Gaussian noise added. We take $N_i = 2N_0$; though this choice may seem rather contrived, it enables us to equalize computational complexities across the two methods, via (39). Different interference conditions are simulated by adjusting the noise floor threshold during peak selection.

By varying the noise floor threshold from 0.05 down to 0.0144 of the maximum STFT amplitude, from three to seven output peaks are accepted. In the three-peak case (threshold = 0.05), no peaks are spurious. In the five-peak case (threshold = 0.03), one low-frequency peak is spurious, and in the seven-peak case (threshold = 0.0144), two become spurious.

In the absence of spurious peaks, Goldstein's method exhibits impressive performance, as Figure 11 displays. Unfortunately, as shown in Figure 12, for the five-peak case, a single low-frequency, spurious peak may utterly destroy the viability of the Goldstein's f_0 estimate. Figure 13 displays results for the seven-peak case, in which two of seven peaks are spurious; no improvement is evident. By contrast, our fast approximate MCMC enumeration maintains a viable f_0 estimate under identical conditions, as shown in Figure 14.

Lastly, we have shown that our MCMC approximation obtains virtually identical results to that of our exact enumeration, while significantly reducing the computational costs induced by the traversal of descriptor states. It is plausible that Goldstein's method admits a similar MCMC traversal strategy, especially when recast in the generalized likelihood evaluation framework outlined in this Appendix. The latter investigation awaits further study.

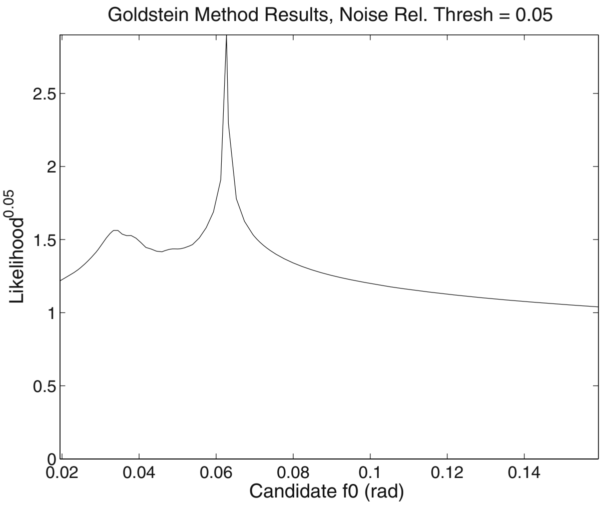


Fig. 11. *Goldstein's method likelihood surface: 3 peaks; 0 spurious*

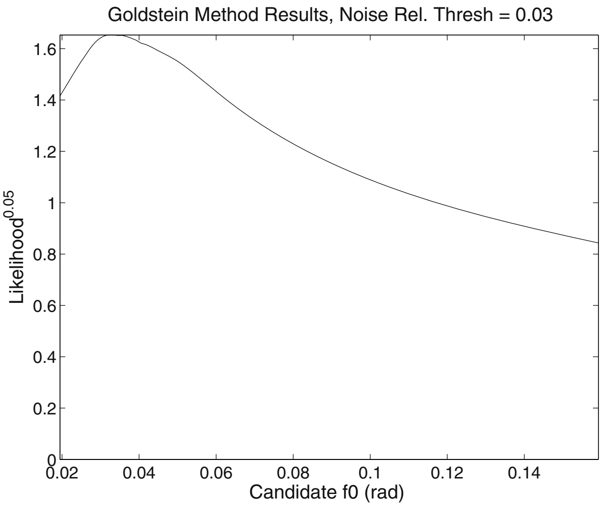


Fig. 12. *Goldstein's method likelihood surface: 5 peaks; 1 spurious*

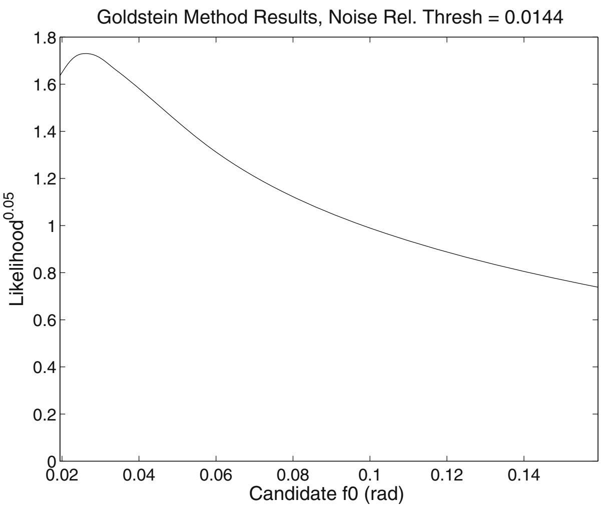


Fig. 13. *Goldstein's method likelihood surface: 7 peaks; 2 spurious*

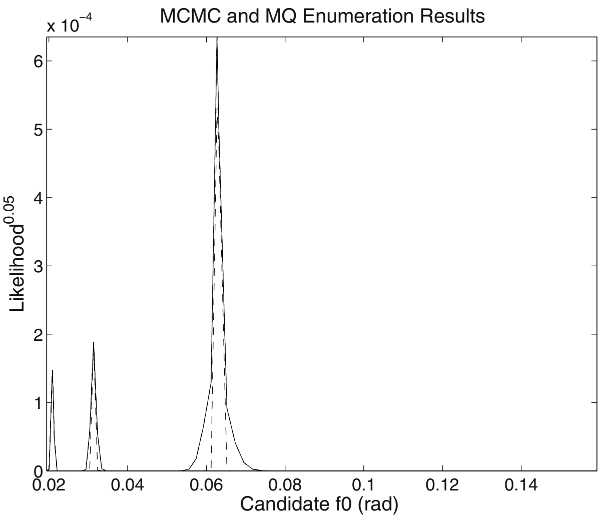


Fig. 14. *Thornburg-Leistikow method, approximate MCMC enumeration (solid line): 7 peaks, 2 spurious*

Determination of Perceptual Tempo of Music

Bee Yong Chua and Guojun Lu

Gippsland School of Computing and Information Technology
Monash University
Churchill, Victoria 3842, Australia
{bee.yong.chua, guojun.lu}@infotech.monash.edu.au

Abstract. *Perceptual Tempo* refers to listener's tempo perception as fast, moderate or slow, when he listens to a piece of music with fairly constant overall tempo. Music perceived to be faster will have higher Perceptual tempo than music perceived to be slower. Existing work on using computer system to automatically estimate the tempo of a piece of music mainly focuses on estimating the Score tempo or estimating the 'Foot-tapping' tempo. However, the existing work is usually not able to determine the *Perceptual Tempo*. This paper proposes an approach to determine this *Perceptual Tempo*. Experimental results show that our proposed algorithm is effective in estimating the *Perceptual Tempo*.

1 Introduction

Rhythm is referred as a temporal pattern with durational and accentual relationships and possibly structural interpretations [2]. Rhythm forms an important part in music. One of the important features of rhythm is the beat, which is defined as a perceived pulse marking off equal durational units [2]. The rate at which beats occur is referred to as tempo, which is often expressed as the number of beats per minutes (bpm).

When listening to a piece of music with fairly constant overall tempo, listeners are able to tell easily whether it is fast, moderate or slow. In addition, between two pieces of music with different overall tempo, listeners are able to decide easily which is faster or slower. We will term this high-level tempo as *Perceptual tempo*. Thus, Perceptual tempo represents the tempo as perceived by listener. Music perceived to be faster will have higher Perceptual tempo than music perceived to be slower.

Existing work on using computer system to automatically estimate the tempo of a piece of music mainly focuses on estimating the Score tempo or estimating the 'Foot-tapping' tempo. Score tempo refers to the tempo reflected in the music score for musicians to follow as they played the music. Whereas Foot-tapping tempo refers to the tempo listener sub-consciously tap along when he listens to a piece of music. However, both the Score tempo and Foot-tapping tempo are usually not the same as the Perceptual tempo. Figure 1 shows one example where the two different music extracts have the same score tempo but different perceived tempo. Based on the estimated Score tempo, the computer system will interpret that these two music extracts have the same tempo. But the actual perceived tempo of the second music extract is about three times slower than the first music extract. As for Foot-tapping tempo, it does not usually correspond to Perceptual tempo because Foot-tapping tempo is centered around average human normal heart beat rate of about 80 to 100 bpm. Thus, for a piece of music with very fast tempo, foot-tapping tempo is usually half of the actual perceived tempo. Table 1 illustrates three pieces of music with different listener's tempo perception but having the same foot-tapping tempo. Based on these Foot-tapping tempos, the computer system will interpret that these three music pieces have the same tempo. However, the actual perceived tempos are different. To date, there is no work done on estimating the Perceptual tempo.

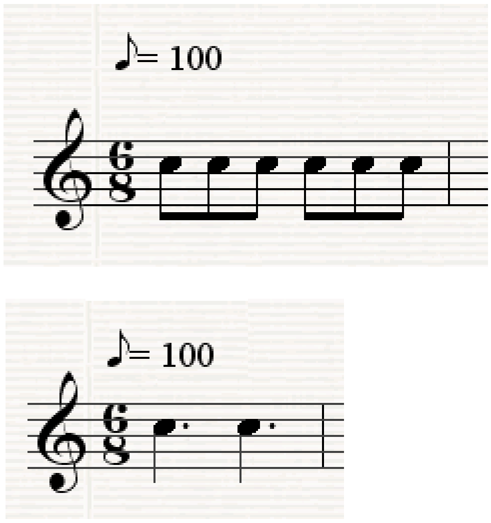


Fig. 1. Illustration of two different music extracts with same Score Tempo

Table 1. Illustration of three Different listener's tempo peception with the Same Foot-tapping tempo

Listener's Tempo Perception	Foot-Tapping Tempo (bpm)	Perceptual Tempo (bpm)
Fast	80	160
Moderate	80	80
Slow	80	40

For most of the music pieces, the Perceptual tempo perceived by listeners with and without music knowledge is usually the same. However for some music pieces, Perceptual tempo may be perceived differently by listeners with music knowledge and by those without. This paper proposes an approach to determine the Perceptual tempo without applying any music knowledge.

There are various applications in determining the Perceptual Tempo. One of them is that it enables the music retrieval system to retrieve music based on listener's tempo perception as fast, moderate or slow. In addition, one of the recent work in music information retrieval is to classify and retrieve music signal based on different emotion expressions in music. However, the retrieval result obtained to date is either based on too broad classification of emotion or the result is not satisfactory [8,11,12]. Perceptual tempo is one of the main factor that enable listener to identify and classify different emotion expression in music [7]. Further, in music psychology research, the empirical findings on how the tempo affects different emotion in music are always in terms of Perceptual tempo. Some of these research work are [1,5,6,9]. Thus, instead of using Score or Foot-tapping tempo, using Perceptual tempo as one of the main features in Emotion-Based Music Retrieval System will defintely aid in getting better retrieval result. Similarly, determination of Perceptual tempo will also aid in retrieving music signal based on different music genre.

The next section will describe and assess the related work in estimating tempo. Following that, we will discuss our proposed approach and outline our proposed algorithm. The subsequent sections will describe the test music data used and the experimental results.

2 Related Work

The common approach in estimating the tempo of a music signal consists of three stages: frequency analysis, amplitude envelops extraction and tempo analysis, as shown in Fig. 2. Among different techniques, the main difference is in the tempo analysis stage whereas the first two stages are rather similar.

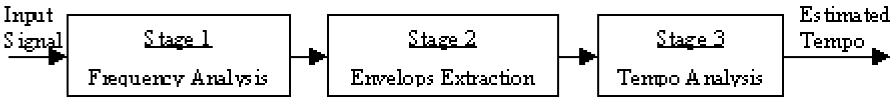


Fig. 2. Common approach in estimating the tempo of a music signal

In frequency analysis stage, the incoming music signal is divided into various frequency sub-bands. The number of sub-bands divided varies in different tempo estimation techniques. At stage 2, the signal in each sub-band is smoothed to produce amplitude envelops for each sub-band. At stage 3, the tempo analysis algorithm estimates the tempo of the music signal from the smoothed amplitude envelops. The tempo analysis algorithms proposed by previous researchers can be classified into event-based and self-similarity based.

2.1 Event-Based Approach

In event-based approach, the onset of the signal is estimated based on the presence of rapid rise in the amplitude envelops. Through the computation of the Inter-Onset-Interval (IOI), multiple beat hypotheses are generated. As accurate extraction of onset times in polyphonic music is still an unsolved problem, the onset detected in the event-based approach is just a rough estimate. And the event-based approach has to be based on the assumption that beat information can be obtained from onset event. Furthermore, in order to estimate the tempo, some other assumptions had to be made. Goto's algorithm only analyzed music with 4/4 timing, with particular style of music and with tempo range between 61 and 120 quarter notes per minutes [4]. In addition, he assumed that the chord changes would occur in strong metrical position. Laroche's algorithm had to limit the estimate tempo to 70 to 140 quarter notes per minutes to avoid halving or doubling the estimate tempo [10]. Further, his algorithm only analyzed music with constant tempo and he assumed that the onset will tend to occur on the first quarter-beat or on the third one but will less often occur on the other quarter-beats. Whereas Seppanen's algorithm assumed that beats correspond to accentuated notes [14].

2.2 Self-Similarity Approach

To detect periodicity, one of the techniques of the self-similarity approach uses comb filters [13]. Comb filters are a set of filters with different delays that cover the range of possible tempo. In this technique, a set of comb filters is applied to the amplitude envelops and the estimated tempo is based on the comb filter that produced the highest response.

Another technique of the self-similarity approach to detect periodicity uses Autocorrelation Function (ACF). ACF compares one signal with the delayed signal of itself. The ACF plot is plotted with the degree of similarity between the signal and the delayed signal against time lag. At time lag zero, it will have the highest similarity and thus it also represents the total energy of the signal. The next highest peak away from time lag zero will reflect the starting time where the rhythm pattern repeats. The peaks reflect the beats occurrence at different duration. Thus, the duration and intensity of each beat occurrence is reflected in the time lag and amplitude of each peak respectively.

There are few approaches in using ACF for tempo analysis. Foote et al. built a beat spectrum using the highest peak found in ACF. The peaks in his beat spectrum correspond to the repetitions in the audio [3]. Tzanetakis' detected the main tempos of the signal by accumulating the dominant peaks found in the ACF of each segment over the whole music signal into a beat histogram. The highest two peaks from the histogram reflect the two main tempos of the signal [15]. However, till present, there is no work done in finding the best peak in ACF plot to best represent the Perceptual Tempo.

Two recent works, one using comb filtering and the other using ACF techniques are Scheirer's and Tzanetakis's algorithm respectively [13,15]. The following sub-sections will explain their proposed algorithm and discuss the weakness of these two techniques in determining the Perceptual Tempo. We then propose an improved tempo estimation technique based on these two techniques.

Scheirer's Algorithm. As in the common approach in estimating tempo, the input signal is first divided into six sub-bands and in each sub-band, the signal is smoothed to produce amplitude envelops. A set of 100 comb filters is then applied to the amplitude envelops in each sub-band. The output of each comb filter is then summed across all sub-bands and the estimated tempo is based on the output filter with the highest summed energy.

The weaknesses in using Scheirer's algorithm to determine Perceptual Tempo are firstly, his comb filtering technique could not detect amplitude envelops with small amplitude variation. Thus, for music signal with small amplitude variation, his algorithm is not able to determine the tempo accurately. Secondly, by summing the output of each comb filter across all sub-bands, the amplitude envelops with higher amplitudes (strong beats) in individual bands are emphasized and thus those lower amplitudes (weak beats) are indirectly de-emphasized. Therefore, for music signal with alternate strong and weak beats rhythm, although the listener is able to perceive the weak beats, the strong beat will dominate the summed output energy across all sub-bands. This leads to his algorithm estimates the tempo lower than the actual perceived tempo. And finally by using the comb filters, it is more computational expensive than ACF technique.

Tzanetakis' Algorithm. In this algorithm, the input signal is divided into different time segments. In each time segment, the segmented signal is divided into different sub-bands and similarly, each sub-band is smoothed to produce amplitude envelops. The amplitude envelops for each sub-band are then normalized and the normalized envelops are then summed across all the sub-bands. In analysing the tempo, ACF is computed from the normalized summed signal and the dominant peaks are then extracted from the ACF plot. The dominant peaks are determined by first removing repetitive peaks in the ACF plot and the first three highest peaks from remaining peaks in the ACF plot are then selected as the dominant peaks. The dominant peaks found in each time segment are then accumulated over the whole music file into a beat histogram. The highest two peaks in the beat histogram are taken as the two main tempo of the music signal.

There are a number of weaknesses in using Tzanetakis' algorithm to determine Perceptual Tempo. First, by taking the highest two peaks in the beat histogram as two main tempos, his algorithm did not determine the actual tempo of the music signal. Second, by normalizing the amplitude envelops, music signal with some beats intensities that are insignificant to listener's perception is amplified. Thus, for this type of music signal, his algorithm is not able to determine the Perceptual Tempo. Third, by summing the energy across all sub-bands, the strong beats are emphasized and the weak beats are indirectly de-emphasized. As in one of the weakness in Scherier's algorithm, for music signal with alternate strong and weak beats rhythm, although the listener is able to perceived the weak beats, the strong beat will dominate the summed output energy across all sub-bands. This leads to Tzanetakis' algorithm estimates the tempo lower than the actual perceived tempo. And finally, he selected the dominant peaks in the ACF plot as the main tempos of segmented signal. However, the dominant peaks found are often not corresponding to the Perceptual Tempo.

3 Proposed Approach

Our hypothesis of Perceptual Tempo is that it corresponds to the most salient event among various music events in a piece of music. Thus, it should correspond to the listener's perception of the fastest tempo, above certain intensity level, among various music events in a piece of music. In ACF plot, this tempo should be reflected in the first event, or the shortest time lag, with the first peak exceeding certain threshold.

Thus, we determine the Perceptual tempo by applying the above hypothesis and by overcoming the weaknesses found in Scheirer's and Tzanetaks'. The following measures are carried out:

- Overcome the comb filtering problem by using ACF
- Overcome the normalization problem in Tzanetakis' by analysing the tempo from Un-normalized amplitude envelop.
- Overcome the summing across sub-bands problems by first estimating the tempo in each sub-band. Instead of estimating the tempo by selecting the dominant peaks found in the ACF plot, our proposed algorithm estimates the tempo by applying the above hypothesis.

The following are the details of our proposed algorithm:

- As in the common approach, the input signal is first divided into six sub-bands. Through empirical finding,we found that six sub-bands are sufficient for our algorithm to determine the Perceptual Tempo.
- As in stage 2 of the common approach, in each sub-band, the signal is smoothed to produce amplitude envelops.
- In analyzing the tempo, ACF is computed on each un-normalized smoothed sub-band rather than on the normalized summed amplitude envelops across all sub-bands as in Tzanetakis' algorithm.
- In each sub-band, the peak in the ACF plot with the shortest time lag and with peak magnitude above 20% of the total energy (peak magnitude at zero time lag) is selected. This threshold of 20% was determined empirically.
- The Perceptual Tempo of the music signal is then determined by finding the shortest time lag among the selected peaks.

4 Experimental Results

4.1 Test Data

The test data consists of 25 music extracts, where 5 are converted to WAV format from MIDI and the rest extracted from CD recording and converted to

WAV format. All the 25 music extracts are in 16-bit mono audio with sampling frequency rate of 11025Hz. Duration of the music extracts is mostly the first 1 minutes of the music with a few exceptions with the first 30 seconds of the music.

Each of the music extracts has fairly constant overall perceived tempo. In all these extracts chosen, the tempo perceived by both listeners with and without music knowledge is the same. The test data represents different listener's tempo perception. Some have very fast tempo, about 180 bpm, some moderate tempo and some have very slow tempo, about 50 bpm. In addition, test data are made up of a number of different musical genres, which include ballad, rock, popular music with vocals and with and without drumming, instrumental with and without drumming, classical without drumming, jazz and blues.

4.2 Results

In Table 2, column 2 reflects the Ground truth or the listener's tempo perception. The Ground Truth is determined manually by tapping how fast or slow a piece of test music is perceived. The tapping is based on the most salient event in the music piece.

The experimental results obtained are in bpm and results in Bold and Italics indicate the tempos that are not corresponding to Perceptual Tempo, or Ground Truth in column 2. Due to expressive timing in music, it is rare to find music pieces with fixed tempo throughout, although listener may perceived it to be fairly constant. But rather, for music with fairly constant perceived tempo, the tempo is usually vary slightly throughout the whole music piece. Therefore, if the result obtained is slightly different from the Ground truth, it is considered to be corresponding to the Ground truth.

Column 3 is the results obtained by running Scheirer's codes. Column 4 is the results obtained by running Tzanetakis' codes. It indicates the 2 highest peaks in the beat histogram generated from his code, with the left figure being the highest and the right the next highest peak.

The last column is the estimated tempo generated from our proposed algorithm. The table shows that the tempos estimated from our proposed algorithm are the closest to the ground truth. This implies that our algorithm is effective in determining the Perceptual Tempo of the music signal.

5 Discussion

In Table 2, almost half of the results obtained from Scheirer's and most of the results from Tzanetakis' algorithm shows multiples or fractions of the ground truth.

In Scheirer's results, more than one-third of the 25 music extracts used were either multiples or faction of the ground truth. Ballad 3, Popular 1 and Classical 1 to 4 have amplitude envelops with small amplitude variation. As discussed earlier, Scheirer's algorithm had problem in placing the correct perceived beat

Table 2. Experimental Results. See Sect. 4.2 for description.

Genre	Ground Truth	Scheirer's	Tzanetaks'	Proposed Approach
Ballad 1	73	***146***	72 / 45	73
Ballad 2	70	69	***191*** / ***137***	69
Ballad 3	76	***154***	***42*** / ***101***	77
Rock 1	102	101	***50*** / 100	101
Popular 1	55	***100***	49 / ***193***	52
Popular 2	170	165	***40*** / ***81***	163
Popular 3	69	69	***258*** / ***139***	69
Popular 4	64	64	63 / ***127***	64
Popular 5	140	136	***268*** / 134	134
Popular 6	132	***66***	***66*** / 132	133
Popular 7	71	69	69 / ***178***	69
Popular 8	150	151	***75*** / ***99***	151
Popular 9	72	66	66 / ***126***	88
Instrumental 1	170	***86***	***56*** / 166	170
Instrumental 2	150	154	***60*** / 151	153
Instrumental 3	175	***89***	***44*** / ***89***	180
Instrumental 4	70	71	70 / ***47***	71
Instrumental 5	110	108	108 / ***53***	108
Classical 1	56	***165***	***132*** / ***101***	57
Classical 2	60	***126***	41 / ***246***	59
Classical 3	55	***165***	***268*** / ***258***	32
Classical 4	64	***117***	***191*** / 60	66
Jazz 1	105	109	***46*** / ***72***	109
Jazz 2	121	121	***60*** / 161	121
Blues 1	67	66	***139*** / 67	65

with this type of amplitude envelops. Thus, the estimated tempo for these music extracts was at least double of the ground truth tempo. Further, Popular 6, Instrumental 1 and 3 are music extracts with very fast perceived tempo and they all have rhythm with alternate strong and weak beats. As discussed earlier, Scheirer's algorithm summed the energy across all sub-bands. This emphasized the strong beats intensity in each sub-band. Therefore, although listener is able to perceived the weak beats clearly, Scheirer's algorithm can only detect the estimated tempo as half of the ground truth.

In Tzanetaks' results, he did not determine which is the main tempo between the two highest peaks selected. Thus, there is no conclusion into which is the actual tempo estimated for each music extract. In addition, most of the results obtained were not correspond to the ground truth. As discussed earlier, this is because the ACF computation is based on the normalized summed amplitude envelops across all sub-bands and that the dominant peaks in the ACF plot are taken as the main tempos of the segmented signal.

Our proposed method applies our hypothesis of Perceptual Tempo and overcomes the weaknesses found in Scheirer's and Tzanetaks' tempo estimation algorithm. Thus, as shown in Table 2, the results obtained from our proposed approach are close to the ground truth. This implies that our proposed approach is effective in determining the Perceptual Tempo of the music signal.

6 Conclusion

In this paper, we proposed an approach to estimate the Perceptual tempo. Our hypothesis of Perceptual Tempo is that it corresponds to the most salient event among various music events in a piece of music. Thus, we determine the Perceptual tempo by applying the above hypothesis and by overcoming the weaknesses found in existing work. Experimental results show that our proposed algorithm is effective in estimating the Perceptual Tempo.

Advantages of our proposed approach are:

– It is simple and computation efficient way of determining the Perceptual Tempo.
– It is more reliable than existing work in determining the Perceptual Tempo.
– This algorithm does not require prior knowledge about the music signal, like musical timbre, genres or even notes or onsets.
– It does not limit to estimating tempo with only 4/4 timing or with particular style of music.

References

1. Balkwill, L.L., Thompson, W.F. A cross-cultural investigation of the perception of emotion in music: Psychophysical and cultural cues. Music Perception, vol. 17, (1999), 43-64.
2. Dowling, W. J., Harwood, D.L. Music Cognition. Academic Press, London (1986).
3. Foote, J., Uchihashi, S. The Beat Spectrum: A New Approach to Rhythm Analysis. IEEE International Conference on Multimedia and Expo, Tokyo, Japan (2001).
4. Goto, M. An Audio-based Real-time Beat Tracking System for Music With or Without Drum-sounds. Journal of New Music Research vol.30 issue 2 (2001) 159-171.
5. Gundlach, R.H. Factors determining the characterization of musical phrases. American Journal of Psychology, vol. 47, (1935), 624-644.
6. Hevner, K. The affective value of pitch and tempo in music. American Journal of Psychology, vol. 49, (1932) 621-630.
7. Juslin, P.N., Sloboda, J.A. Music and Emotion: Theory and research. Oxford University Press (2001).
8. Katayose, H., Inokuchi, S. The Kansei music system. Computer Music Journal, vol. 13(4), (1989), 72-77.
9. Krumhansl, C.L. An exploratory study of musical emotions and psychophysiology. Canadian Journal of Experimental Psychology, vol. 51, (1997) 336-352.
10. Laroche J. Estimating tempo, swing and beat locations in audio recordings. IEEE Workshop on Application of Signal Processings to Audio and Acoustic (WASPAA), New Paltz, New York, USA (2001) 135-138.

11. Li, T., Ogihara, M. Detecting Emotion in Music. International Symposium on Music Information Retrieval (ISMIR), Baltimore, Maryland, USA (2003).
12. Liu, D., Lu, L., Zhang, H. Automatic Mood Detection from Acoustic Music Data. International Symposium on Music Information Retrieval (ISMIR), Baltimore, Maryland, USA (2003).
13. Scheirer, E. Music Listening System. Doctor of Philosophy, Massachusetts Institute of Technology (2000).
14. Seppanen, J. Computational Models of Musical Meter Recognition. Master of Science Thesis, Tampere University of Technology (2001).
15. Tzanetakis, G. Manipulation, Analysis and Retrieval Systems For Audio Signals. Doctor of Philosophy, Princeton University (2002).

Source Separation and Beat Tracking: A System Approach to the Development of a Robust Audio-to-Score System

Mario Malcangi

LIM – Laboratorio di Informatica Musicale
DICo – Dipartimento di Informatica e Comunicazione
Università degli Studi di Milano
Via Comelico 39, 20135 Milano, Italy
malcangi@dico.unimi.it

Abstract. A set of tools for pitch, metric and rhythm information matching on musical audio streams are under development. The goal is a robust automatic system capable to track musical audio data for information retrieval and access purpose. Several problems have been identified and approached to be solved separately, such as metric computation, rhythm recognition and pitch tracking. This approach has been chosen to synthesize a whole processing model robust enough to be applied virtually to any kind of music. A source separation modeling has been investigated starting from ICA model. An ICA modified unmixing model has been proposed as preprocessing subsystem. This subsystem demonstrated to be very helpful for efficient applying of processing algorithms to rhythm and pitch tracking.

1 Introduction

Our research aims at automatic recognition of musical information in audio streams and to gain the corresponding symbolic representation. The goal is very ambitious as automatic recognition of musical information in audio streams is currently one of the most complex pattern recognition task, more than speaker-independent vocabulary-unlimited continuous speech recognition [1]. The reason is in the high degree of complexity and variability of musical information related to its signal nature.

The field of automatic transcription of musical sounds has been explored since the beginning of 1970s, when some first basic researches, such as the one on segmentation and analysis of sounds by digital computer by James A. Moorer at Standford University [2]. Several important results has been gained in the last decades, above all when non traditional signal processing and patterns matching methodologies has been successfully used to solve some non linear problems, such as independent audio sources separation.

Today the goal is larger than the one concerning a niche application such as automatic music transcription. A larger goal is the development of automatic musical sound information recognition system targeted to automate processes like information retrieval on audio media and others similar applications.

Beat, rhythm and pitch tracking on audio are three rather distinct tasks, all essential for a complete and efficient automatic musical sound transcription system. Each of those may be simple and successful tasks if we refers to electronically-generated sounds like amplitude modulated sinusoids, but became complex and faulty tasks when applied to an orchestra symphony performance.

We propose a system approach that decomposes the whole problem in low complex sub problems and lowers the complexity of audio information source. Independent source separation has been identified in our researches as a primary goal to be gained prior to approach the solution of the general problem of audio information matching and corresponding symbolic music scoring.

The proposed Audio-To-Score system development approach consists of a preprocessing subsystem for sources separation, a set of processing subsystem targeted to beat, rhythm and pitch tracking, and a soft-computing based post-processing subsystem for decision on uncertain data.

Source separation and beat tracking have been developed and tested. Rhythm, pitch and soft-computing based decision logic for score synthesis is under development.

2 Sources Separation

Audio streams contain a mixture of sounds from different sources. Auditory Scene Analysis (ASA) [3] is the process that leads to individual source extraction by a hearing system based on localization, denoising and matching. The Computational Auditory Scene Analysis (CASA) [4] is a computer-based implementation of ASA for extraction of individual audio sources from a mixture of microphone-recorded multiple sounds.

CASA is ideally a good approach to solve the source separation problem but for an effective implementation it need to embed substantial knowledge of human auditory system and psychophysical behavior [5]. Multiple microphone unmixing algorithms, such as ICA (Independent Component Analysis), are to be considered a good compromise solution to the source separation problem as they are blind methods. ICA works very well on mixture of independently recorded sound sources as it is able to uncorrelate sources altered by acoustic environmental characteristics (delay and reflections) without any a-priori model. If an N-microphone N-sources is available, ICA is able to separate them very efficiently [6]. Unmixing schema is internally estimated by ICA algorithm on a statistical base. Fast ICA algorithm implementation is also available for real-time applications [7].

Unfortunately ICA fails to unmix one-microphone recorded sound sources. In this case, the most recurrent, environmental characterization is not dominant, and different sound sources are not available to feed the input of this algorithm. This problem became harder to solve using an ICA-based approach if multiple sources are generated from a single instrument playing a multiple voice musical score, such as a piano played polyphonic song.

Anyway ICA unmixing is potentially a good approach the solution of the complex problem of audio source separation. Original ICA algorithm can be successfully used to de-emphasize interferences among individual sound sources, then a modified version, refiltering [8], can be applied to each unmixed sound source to derive separate audio tracks where polyphony occurs.

Refiltering consists in constructing new sources by means of a selectively reweighting of multiband-splitted signals. Weights are variable over time and act as masking signals:

$$s(t)=a_1(t)b_1(t)+ a_2(t)b_2(t)+ a_3(t)b_3(t)+\ldots+ a_m(t)b_m(t) \quad \textbf{(1)}$$

$s(t)$: signal to be estimated
$a_i(t)$: i-th reweighting coefficient
$b_i(t)$: i-th original source sub-band.

One-microphone source separation using refiltering is possible if masking signals are optimally chosen and the whole process can be automated.

2.1 ICA Unmixing Model

Developed to solve the “cocktail party problem”, the Independent Component Analysis (ICA) suits very well to unmix sound sources recorded by a set of independent microphones.

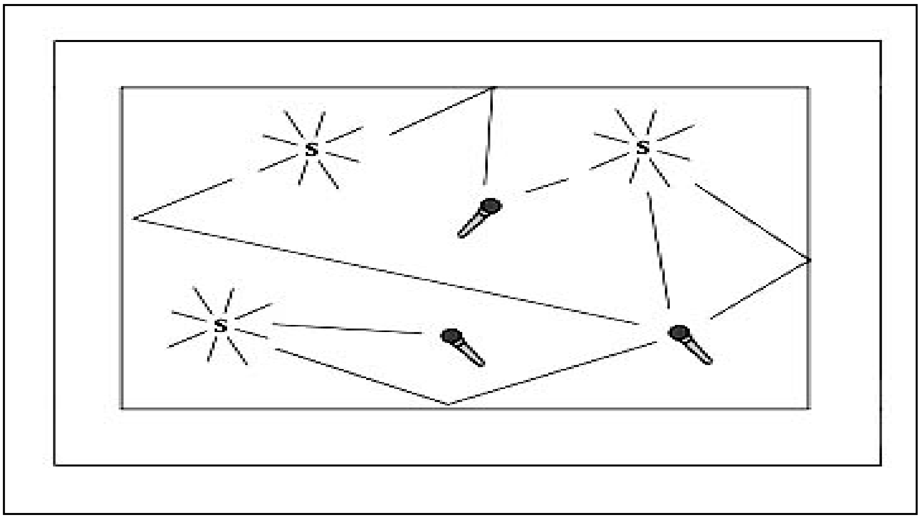


Fig. 1. Multiple sources can be separated if each one is captured by an independent microphone.

Main property of ICA algorithm is its great efficiency in sources separation when high statistical independence occurs among original sources and a mixture of them is available:

$$x_j = a_{j1}s_1 + a_{j2}s_2 + \ldots + a_{jn}s_n \, . \tag{2}$$

Using array notation, mixing model can be described as:

$$x = As \, . \tag{3}$$

ICA uses statistical independency propriety of the sound sources to estimates the unknown mixing matrix A. After A has been estimated, its inverse W can be computed so that sound sources can be separated:

$$s = Wx \, . \tag{4}$$

Sound sources are not effectively computed but estimated, as the mixing matrix A has been estimated without any real knowledge about effective sound sources s. ICA is a *Blind Source Separation* (BSS) method.

Statistical independency of the sound sources is a fundamental condition for a successful ICA application. This is an applicable hypothesis for musical sound sources, so that ICA can be applied successfully for this purpose.

2.2 ICA Modified Model

ICA unmixing processing model cannot be applied if we haven't the N microphone recording of the N sound sources. This happens in almost the totality of the cases, as only final mastered version of musical execution is available (e.g. stereophonic recording of an orchestra performance).

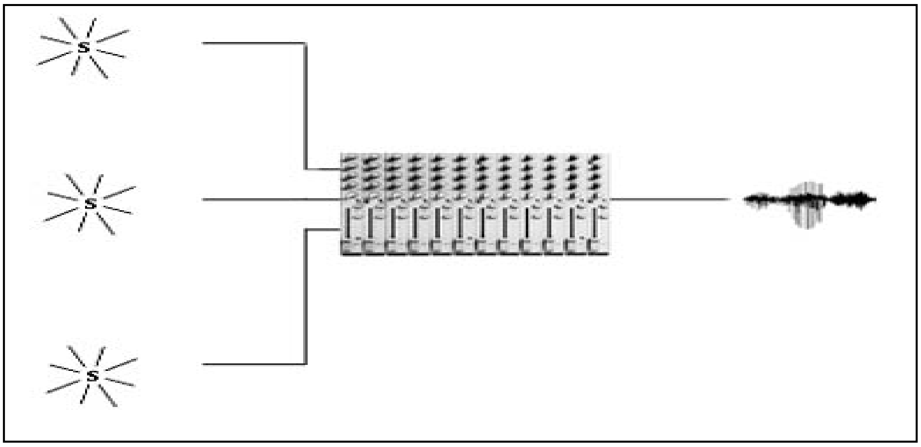


Fig. 2. Multiple sources are generally mastered to a mono or stereo equivalent source.

A good choice to solve the above problem is to consider the audio source as a one-microphone recording of multiple sources. To apply ICA to this case it is necessary to process the one-microphone source so that a multiple-microphone set of sources can be derived [9]. Refiltering can be a successful strategy to do it.

A valid hypothesis is that each sound source carries relatively independent energy information in a narrow band and in a short time interval. A short-term Fourier analysis can be used to buildup a filterbank to be applied to the one-microphone source. We can then synthesize the *N*-microphone sound sources useful to feed ICA algorithm, and to estimate the original independent sources from a one-microphone equivalent source.

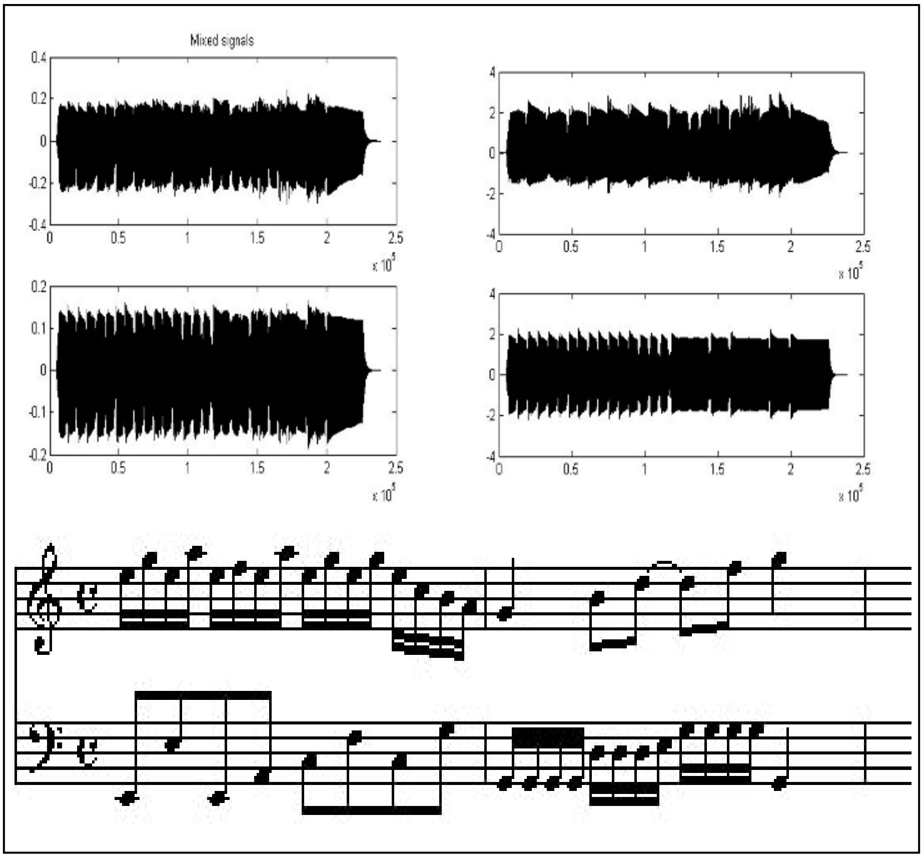


Fig. 3. When pitch sequences are available as single sound streams, metric, rhythm, and pitch extraction can be efficiently executed, as interference among sounds has been minimized.

We propose a dual-step preprocessing of an audio stream. First step consists in applying ICA to separate independent sound sources where multiple-microphone recording is available. Second step consists in a refiltering action applied to the separated independent pitch sequences embedded in each sound source.

3 Metric and Rhythm Estimation

Starting from Scheires's [10] and Sepännen's [11] works, a novel model for metric estimation has been proposed [12].

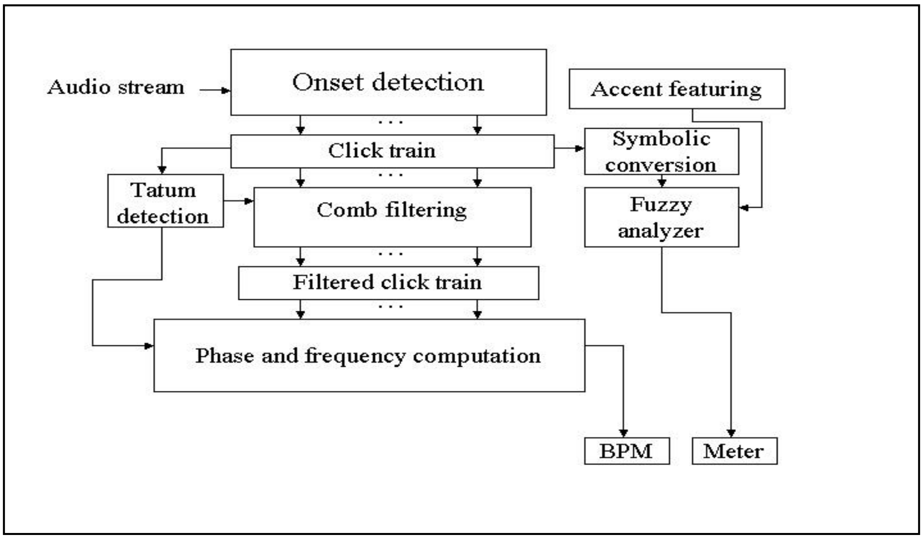


Fig. 4. Metric estimation is a primary task in automatic music transcription as it makes available a set of important timing data concerning the musical information coded in an audio stream.

Audio signal is first processed so that a set of click-train streams is computed. Click-train data are useful for Inter Onset Intervals (IOIs) computation and for attack detection. Resonant filter-bank processing is applied to click-train data so that noisy data can be filtered and only useful data for BPM computation are passed. IOSs information is used as control data for resonant filter-bank setting.

For meter estimation a different processing model has been defined. This is a sofcomputing-based processing (fuzzy rule-based) as linear modeling of this process is too difficult to implement. To feed the fuzzy-logic analyzer engine a symbolic conversion of click-train streams is needed and accent rules need to be available. Fuzzy processor will be trained by means of expert knowledge transfer in terms of inference rules. This part of the whole processing system is under development.

3.1 Onset Detection

To approach automatic beat measuring, audio stream has to be preprocessed so that onset detection can be done with high degree of precision.

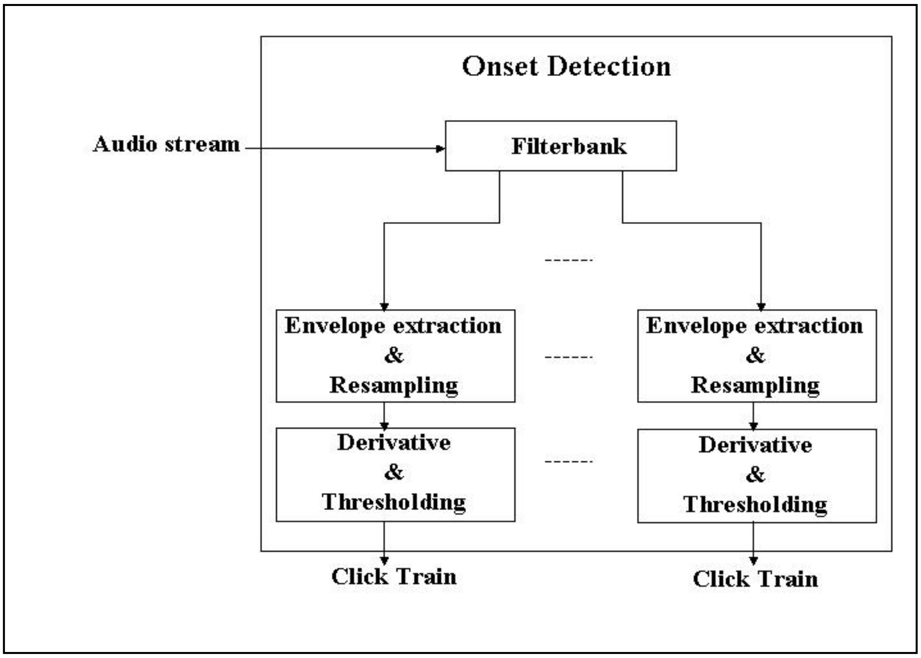

Onset Detection
Audio stream
Filterbank
Envelope extraction & Resampling
Derivative & Thresholding
Envelope extraction & Resampling
Derivative & Thresholding
Click Train
Click Train

Fig. 5. Onset detection processing flow includes filter bank processing and derivative to transform audio signal in a pulse train like sequence.

The first processing step for onset detection is based on filter bank processing. Audio stream has been splitted into N frequency bands as follows:

1. 0 to 200 Hz
2. 200 to 400 Hz
3. 400 to 800 Hz
4. 800 to 1600 Hz
5. 1600 to 3200 Hz
6. 3200 to Nyquist rate

A set of 6th order band-pass elliptic filters has been used to gain enough band separation.

From each narrow band signal the envelope pattern has been computed as follows:

$$Env(n) = g(n) * abs[s_i(n)] \qquad (5)$$

$s_i(n)$ is the narrow band signal

$g(n)$ is the impulse response of a 5 Hz low-pass Butterworth filter.

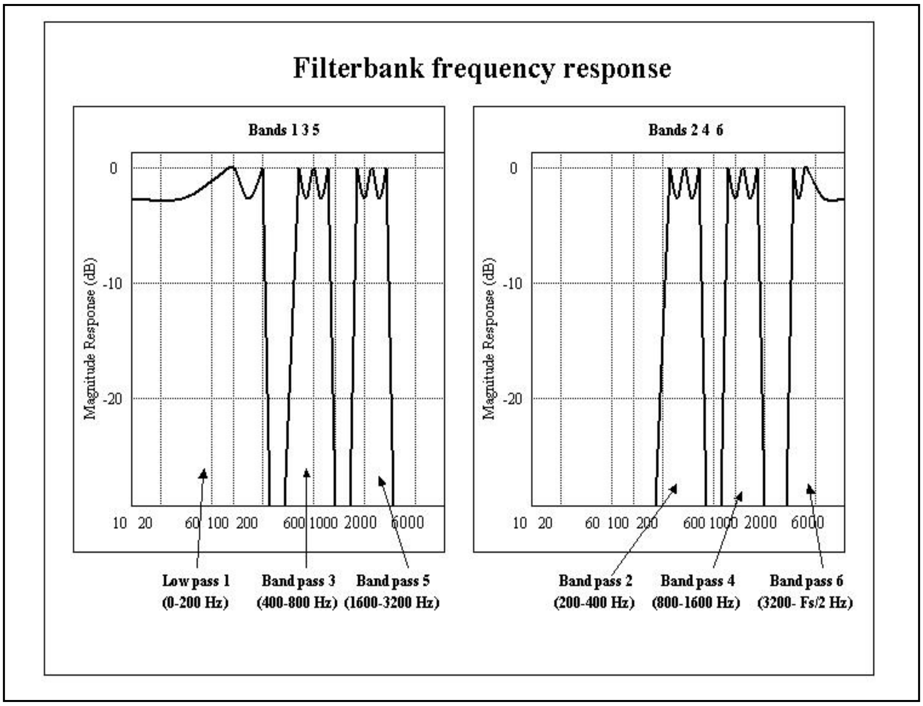


Fig. 6. 6th order band-pass elliptic filters has been used to separate overlapping onsets.

First-order derivative is then applied to each envelope pattern to detect onsets. Derivative has been approximated through the difference.

$$\begin{aligned} &\text{ClickTrain}(n) = 0 && \text{if Env}(n+1)-\text{Env}(n) < k \\ &\text{ClickTrain}(n) = \text{Env}(n+1) && \text{otherwise} \end{aligned} \qquad (6)$$

K : threshold level

3.2 Inter Onset Intervals and Beat Frequency

Given two onsets at times *t1* and *t2*, *t1* < *t2*, Inter Onset Interval (IOI) is a time interval occurring between the onsets and it is defined as *o* = *t2* – *t1*.

After IOIs have been measured, *tatum*, the minimum metric unit, has been computed as the nearest maximum common divisor of IOIs.

This information has been used to drive a resonant filters bank. Comb filtering has been used to this purpose.

Each comb filter have its own resonance frequency. If the frequency of a click train is the same of the filter resonance, filter output signal level is high, otherwise it

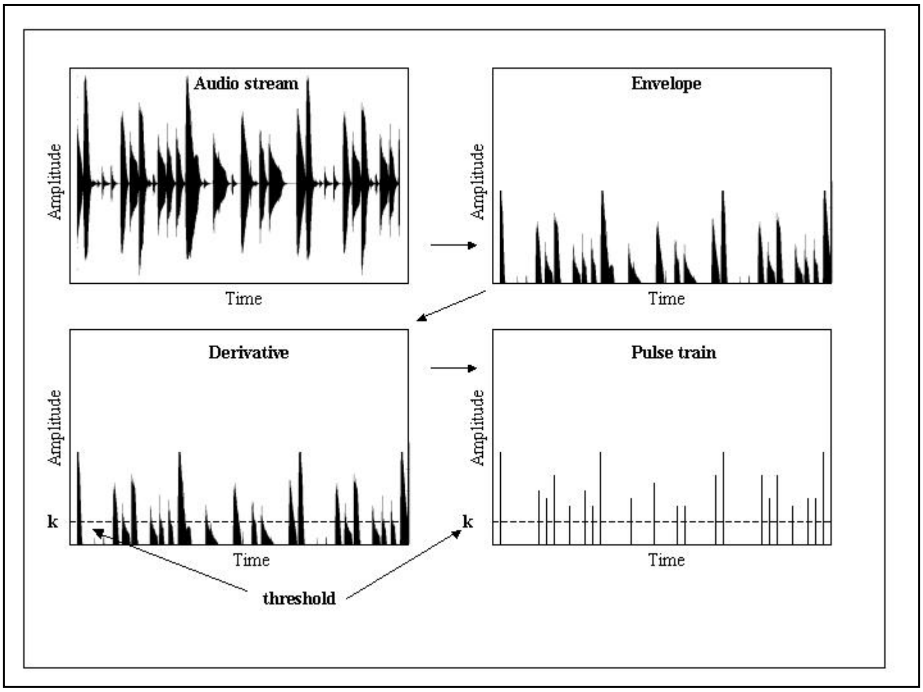


Fig. 7. Thresholding of derivative is useful to clean onset pulse train from background noise.

is low. Comb filters feature also harmonic resonance so that maximum output level happens also at integral multiples of its characteristic frequency.

Comb filters resonance has been computed according to the following formula:

$$\mathbf{energy}(r) \equiv \frac{1}{n} \cdot \sum_{i=0}^{n-1} \sqrt{e} - r[i] \qquad e \equiv \sum_{i=0}^{n-1} r[i]^2 \quad . \tag{7}$$

Delay of the filter with maximum resonance is used to compute beat frequency. BPM (Beat Per Minute) is computed as:

$$(\gamma / \mathrm{fsClick}) * 60 \ . \tag{8}$$

where γ is the maximum resonance comb filter delay, and fsClick is the frequency used to sample the click train envelope.

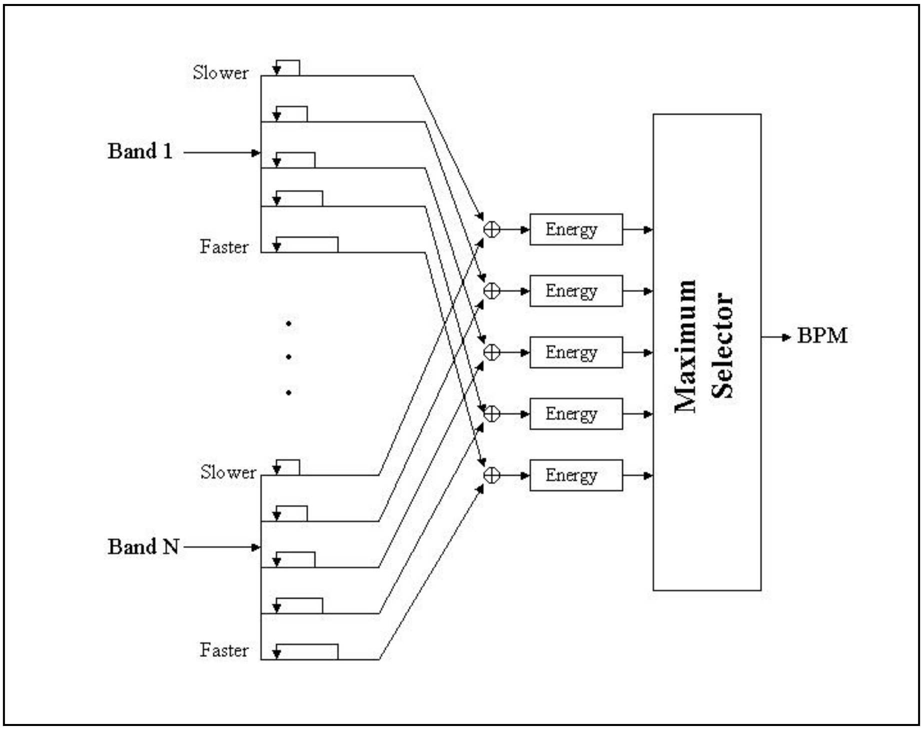


Fig. 8. For each band resonance has been measured sampling the pulse train in a speed range.

3.3 Metrics Estimation

For accent recognition we consider only durational accent. This means that we don’t consider harmonic change, pitch accent and other accent that need a frequency analysis. This will be done later when more timing information has been collected.

After the selection of accent feature we use a fuzzy logic processor to detect accent from the symbolic representation of onset.

Fuzzy logic processing has been introduced at this step, as it is too complex to obtain a linear model of the accent detection. A fuzzy processor enables to transfer the expertise knowledge (musician) into its rules and membership functions.

A fuzzy logic processor is then trained to recognize accents in the audio musical sequence using as input harmonic changes, pitch accent and other extracted information.

The last stage has the task to use the information recognized to find the measure of the musical piece. After the resonant filter bank step, the period of the beat is evaluated selecting the frequency of the resonator that has the maximum peak. Other functions are applied to this step to find the best BPM prediction. The second step has the task to try to find a high level of metrical structure. Accent recognition allows to determinate the meter in a musical piece.

4 Results

Only the part of the whole Audio-To-Score system concerning source separation and beat tracking has been developed and tested to date. To demonstrate the complete automation capability of the implemented subsystem an audio score (baroque dance music) has been processed to compute synchronization data to drive a virtual puppet to execute dance movies animation [12].

Beat tracking processing demonstrated to be very robust for monophonic musical scores but non enough for polyphonic musical scores. If polyphonic audio stream is de-emphasised by means of sources separation, then better performance has been gained. This encourage us to investigate more on source separation as this lead to simplify beat, metric and pitch tracking processing toward an efficient Audio-To-Score system implementation.

Using independent sources separation as a preprocessing front-end of the rhythm and beat tracking process we observe a rise in the performance of the whole system. Noisy audio streams can be processed successfully as sources separation acts as a noise reduction processing. Thresholds can be lowered without increasing of false onset detection rate.

This result is a basic step to setup a framework for rhythm and pitch estimation processing. Beat and rhythm data are to be used to feed a data-driven pattern matching process.

5 Future Framework for Rhythm and Pitch Estimation

If metric is available, rhythm estimation became a simpler task and pitch computing can be droved by the above information to be executed simpler and effectively. Linear signal processing is useful but not enough to avoid artefact and masking effects due to very dynamic and frequency reach audio streams. A mixed-methodology approach is needed to develop a very robust Audio-To-Score system.

We propose a mixed-methodology approach (linear and non-linear processing) to solve higher level information extraction from audio stream. This results in a robust decision system on metric and frequency information coming from linear estimators.

Neural networks modelling can be also used for pattern matching purpose where a physical instrument model is known. This can be more effective for pitch and rhythm tracking when poor sources separation has been executed at pre-processing time.

Acknowledgements

The author wish to acknowledge the partial support of this project by Italian MIUR (FIRB “Web-Minds” project N. RBNE01WEJT_005) and the Italian National Research Council, in the framework of the research program “Methodologies, techniques, and computer tools for the preservation, the structural organization, and the intelligent query of musical audio archives stored on heterogeneous magnetic media”, Finalized Project “Cultural Heritage”, Subproject 3, Topic 3.2, Subtopic 3.2.2, Target 3.2.1. We also want to acknowledge the members of the IEEE MX WG (PAR1599) for their cooperation and interest in our work.

This work has been made possible by the efforts of researchers and graduate students of LIM.

Particular thanks to Goffredo Haus, LIM scientific director, for its encouragement and tips that leaded to set-up this research.

Thanks also to Alessandro Nivuori and to Daniele Murroni for their efforts devolved on some specific topics concerning this paper.

References

1. Malcangi, M.: A mixed methodology speech-to-text recognition system, Proceedings of the twelfth Turkish symposium on Artificial Intelligence and Neural Networks (TAINN-03), Canakkale, Turkey, July 02-04, 2003.
2. Moorer, J. A.: On the segmentation and analysis of continuous musical sound by digital computer, PhD thesis, Dept. of Computer Science, Standford University, May 1975, report number STAN-M-3.
3. Bregman, A. S.: Auditory Scene Analysis,MIT Press, 1994.
4. Brown, G., Cooke, M.: Computational Auditory Scene Analysis, Computer, Speech and Language, 8, 1994.
5. García, L., Casajús-Quirós, J.: Separation of musical instruments based on perceptual and statistical priciples, 2nd COST-G6 Workshop on Digital Audio Effects (DAFx99), Trondheim, Norway, December 1999.
6. Lee, T-W., Bell, A.J., Orglmeister, R.: Blind source separation of real world signals, Proceedings of IEEE International Conference Neural Networks, pp 2129-2135, Houston, June 97.
7. Hyvärinen, A.: Fast and Robust fixed point algorithm for independent component analysis, IEEE Trans on Neural Networks, 10(3): 626-634,1999.
8. Roweis, S.,T.: Factorial models and refiltering for speech separation and denoising, Proceedings of Eurospeech03 (Geneva), pp. 1009—1012, 2003.
9. Cauwenberghs, G.: Monaural separation of independent acoustical components, IEEE Symposium Circuit & Systems (ISCAS’99), 1999.
10. Scheirer, E.: Tempo and Beat analysis of acoustic musical signals. J. Acoust. Soc. Am., 103(1): 588-601, January, 1998.
11. Sepännen, J.: Computational models of musical meter recognition. Master’s thesis, Tampere Univ. of Tech., Tampere, Finland, 2001.
12. Malcangi, M., Nivuori, A.: Beat and rhythm tracking of audio musical signal for dance synchronization of a virtual puppet, Proceedings of the XIV Colloquium on Musical Informatics (XIV CIM 2003), Firenze, May 8-10, 2003.

A Causal Rhythm Grouping

Kristoffer Jensen

Department of Computer Science, University of Copenhagen
Universitetsparken 1, DK-2100 Copenhagen, Denmark
krist@diku.dk

Abstract. This paper presents a method to identify segment boundaries in music. The method is based on a hierarchical model; first a features is measured from the audio, then a measure of rhythm is calculated from the feature (the rhythmogram), the diagonal of a self-similarity matrix is calculated from the rhythmogram, and finally the segment boundaries are found on a smoothed novelty measure, calculated from the diagonal of the self-similarity matrix. All the steps of the model have been accompanied with an informal evaluation, and the final system is tested on a variety of rhythmic songs with good results. The paper introduces a new feature that is shown to work significantly better than previously used features, a robust rhythm model and a robust, relatively cheap method to identify structure from the novelty measure.

1 Introduction

As more and more of the music delivery and playback are made through computers, it has become necessary to introduce computer tools for the common music tasks. This includes, for instance, common tasks such as music playback, control and summary. This paper presents a novel approach to the music segmentation tasks based on rhythm modeling. Music segmentation is here seen as the identification of boundaries between common segments in rhythmic music, such as intro, chorus, verse, etc. These boundaries often consist of changes in the rhythm. The segmentation work undertaken here introduces structure in the music, whereas the previous work [15], on which this work is partly based, mainly investigated the tempo.

The segmentation is useful for many tasks. This approach, which is both real-time and not too processor intensive, is useful in real-time situations. One use is to perform live recomposition, using for instance Pattern Play [20], where the found segments is reintroduced into the music, potentially after some effects performed on the segment. Another use is to assists Djs in computer based DJ software, such as Mixxx [1], for beat mixing, intro skipping, or other uses.

The current approach is built on previous work in beat and tempo estimation [15], where a Beat Histogram was used to estimate the tempo. Only the maximum of the beat histogram was used. In this work, the full histogram is calculated for each time frame. The self-similarity [8, 9] of the histogram, which is here called a rhythmogram, is calculated, and a measure of novelty [9] is extracted. The novelty measure is only calculated on the diagonal of the self-similarity matrix, which thus necessitates only the calculation of a small subset of the full matrix. Finally the segments are found by smoothing the novelty measure, identifying the peaks (the segment boundaries), and following them to the unsmoothed case in several steps using a technique borrowed from edge detection in image scale-space.

Several authors have presented segmentation and visualization of music using a self-similarity matrix [10, 2, 21] with good results. Other methods to segment music include information-theoretic methods [7], or methods inspired from ICA [3].

When designing a music section grouping, or section-clustering algorithm, it is intuitive to try to understand what knowledge there is about how humans go about doing the same task. Desain [6] introduced the decomposable theory of rhythm, in which rhythm is perceived by all note onsets, in what he modeled as essentially an autocorrelation step. Scheirer [23] made some *analysis by synthesis* experiments, and determined that rhythm could not be perceived by amplitude alone, but needed some frequency dependent information, which he constructed using six band-pass filters. No experiments were done using filtered signals, by varying only the filter cutoff frequency. This would make probably the success of one amplitude-based feature, if it were suitably weighted by e.g. an equal loudness contour, or the spectral centroid, which weights higher frequencies higher. Several studies have investigated the influence of timbre on structure. [19] found that timbre did not affect the recognition of familiar melodies, but that it did hurt recognition on non-familiar melodies. McAdams [18], studied contemporary and tonal music, and found that the orchestration affects the perceived similarity of musical segments strongly in some cases. He also found that musically trained listeners find structure through surface features (linked to the instrumentation) whereas untrained listeners focused on more abstract features (melodic contour, rhythm). This helped non-musicians recognize music with a modified timbre (piano and chamber music versions). Deliège and Mélen [5] postulates that music is segmented into sections of varying length using cue abstraction mechanism, and the principle of sameness and difference, and that the organization of the segmentation, reiterated at different hierarchical levels, permits the structure to be grasped. The cues (essentially motifs in classical music, and acoustic, instrumental, or temporal otherwise) act as reference points during long time spans. Deliège and Mélen furthermore illustrate this cue abstraction process through several experiments, finding, among other things, that musicians are more sensitive to structural functions, and that the structuring process is used for remembering, in particular, the first and last segment.

Desain thus inspired the use of an autocorrelation function for the rhythm modeling; Scheirer showed the necessity to model the acoustic signal somehow akin to human perception. For simplicity and processing reasons a scalar feature, which does indeed perform satisfactory, is used in this work Deliège and Mélen inspired the use of a hierarchical model presented here, consisting of a feature, calculated from the acoustic signal, a time varying rhythm abstraction, a self-similarity matrix, and a novelty function extracted from the self-similarity matrix.

This paper is organized in the following manner. Section two presents the beat estimation work that is used to find the optimal feature, and introduces the measure of rhythm, section three presents the self-similarity applied to the rhythm, section four gives an overview of the rhythm grouping in one song. In section 5, an evaluation is performed, and finally there is a conclusion.

2 A Measure of Rhythm

Rhythm estimation is the process of determining the musical rhythm from a representation of music, symbolic or acoustic. The problem of automatically finding the rhythm includes, as a first step, finding the onsets of the notes. This approach is used here to investigate the quality of the audio features. The feature that performs best is furthermore used in the rhythm model.

2.1 Beat and Tempo

The beat in music is often marked by transient sounds, e.g. note onsets of drums or other instrumental sounds. Onset positions may correspond to the position of a beat, while some onsets fall off beat. The onset detection is made using a feature estimated from the audio, which can subsequently be used for the segmentation task. In a previous work [15], the high frequency content was found to perform best, and was used to create a beat histogram to evaluate the beat. Other related works include Goto and Muraoka [11] who presented a beat tracking system, where two features were extracted from the audio based on the frequency band of the snare and bass drum. Later Goto and Muraoka [12] developed a system to perform beat tracking independent of drum sounds, based on detection of chord changes. Scheirer [23] took another approach, by using a non-linear operation of the estimated energy of six band-pass filters as features. The result was combined in a discrete frequency analysis to find the underlying beat. As opposed to the approaches described so far Dixon [7] build a non-causal system, where an amplitude based feature was used as clustering of inter-onset intervals. By evaluating the inter-onset intervals, hypothesis is formed and one is selected as the beat interval. This system also gives successful results on simpler musical structures. Laroche [14] built an offline system, using one features, the energy flux, cross-correlation and dynamic programming, to estimate the time-varying tempo.

2.2 Feature Comparison

There have been a large number of possible features proposed for the tasks and tempo estimation and segmentation. This section introduces a new scalar feature, the Perceptual Spectral Flux, and show that it performs better in note-onset detection than other features.

Apart from the possible vector sets (Chroma, MFCC, PLP, etc), [15] evaluated a number of different scalar features for use in beat estimation systems. The approach was to identifying a large number of audio features, and subsequently evaluating the quality of the features using error measures. A number of music pieces were manually marked, by identifying the note transients, and these marks were used when evaluating the features. In [15], the high frequency content (HFC) [17] was found to perform best. In this work, however, another feature has been evaluated, which performs better than the HFC. This feature, here called the perceptual spectral flux (PSF), is calculated as

$$PSF_n = \sum_{k=1}^{N_b/2} W_B \left(\left(a_k^n\right)^{1/3} - \left(a_k^{n-1}\right)^{1/3} \right), \tag{1}$$

where n is the block index, and N_b is the block size, and a_k is the magnitude of the Short-Time Fourier Transform (STFT), obtained using a hanning window. W_b is the frequency weighting used to obtain a value closer to the human loudness contour, and the power function is used to simulate the intensity-loudness power law. The power function furthermore reduces the random amplitude variations. These two steps are inspired from the PLP front-end [13] used in speech recognition.

The error measures used in the evaluation is the signal to noise ratio (S/N), calculated as the ratio between the sum of the *hills* (corresponding to the peaks and corresponding slopes) of the peaks of the feature under test that are matched to a manual mark to the sum of those that are not, and the matched ratio, calculated as the number of matched peaks, divided by the number of manual marks. The feature peaks are chosen as all local maximums above a given running threshold. As the threshold is increased, the signal to noise increased, whereas the matched ratio decreases. The thresholds necessary to obtain an arbitrary value of 75 % matched peaks (which is possible in almost all cases) are found for all features, and the signal to noise ratio is compared for this threshold. In [15], the high frequency content (HFC) was found to have twice as good S/N ratio as the other measured features. Using the same material, the PSF performs twice as good as the HFC. This can be tentatively explained as, since the HFC weight the high frequency most, it indicates mainly the hihat, and the transient instruments, such as the piano. The spectral flow, with no frequency weighting, essentially favors the low frequencies, since these generally have significantly more energy than the mid, or high frequencies. The PSF weight everything approximately as the human ear, and would then indicate both the high frequency sounds, but also the low frequency sounds, such as the bass, or other instrumental sounds with less transient behavior.

The PSF is calculated on a block of 20 msec., with a step size of 10 msec. An example of the PSF, calculated on an excerpt of Train to Barcelona[1], can be seen in figure 1.

2.3 Rhythmogram

The PSF feature indicates most of the manual marks correctly, but it has many peaks that does not corresponds to note onset, and many note onset does not have a peak in the PSF. In order to get a more robust rhythm feature, the autocorrelation of the feature is now calculated on overlapping blocks of 8 seconds, with half a second overlap. Only the information between zero and two seconds is retained. The autocorrelation is normalized so that autocorrelation at zero lag equals one. This effectively prevents loudness variations to have any influence. Other presented models of rhythm include [21], which uses an FFT on the energy output of the

[1] By Akufen. Appearing on Various - Elektronische Musik - Interkontinental (Traum CD07), December 2001

auditory filterbanks, and [22], whose rhythm patterns consist of the FFT coefficients of the critical band outputs. The autocorrelation has been chosen, instead of the FFT used by the two above-mentioned papers, for two reasons, first, it is believed to be used in the human perception of rhythm [6], and second, it is believed to be more easily understood visually

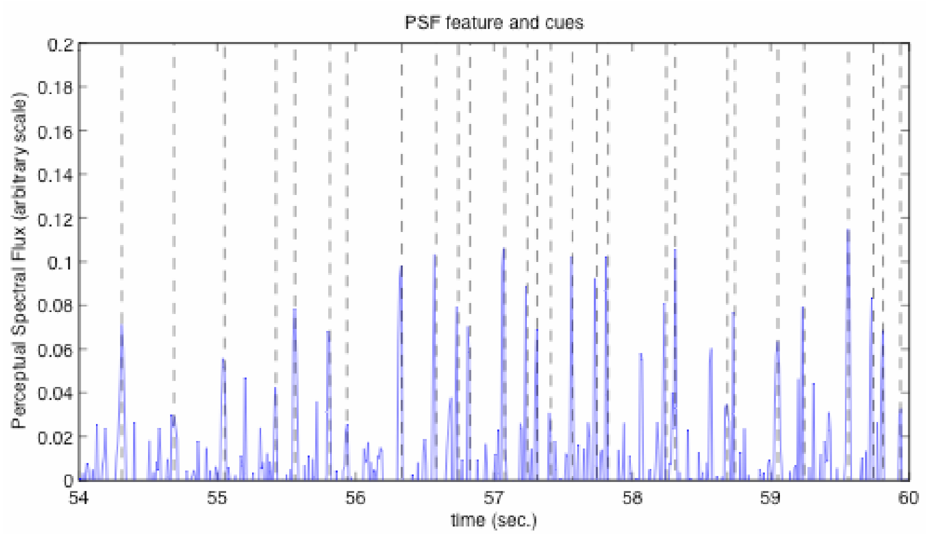


Figure 1. Example of PSF feature, and manually marked note onset marks (dashed vertical lines) for the piece *Train to Barcelona*.

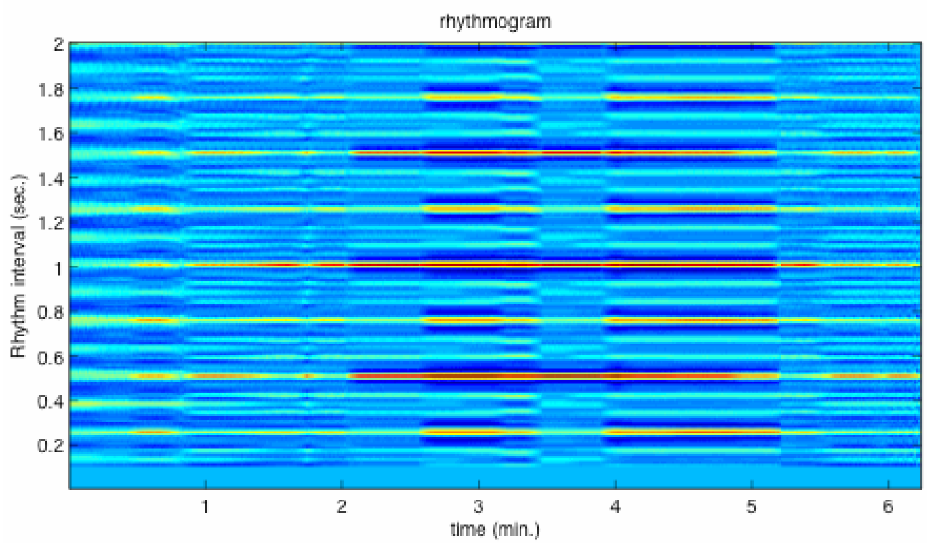


Figure 2. Rhythmogram for *Train to Barcelona*.

If visualized with lag time on the y-axis, time position on the x-axis, and the autocorrelation values visualized as colors, it gives a fast overview of the rhythmic evolution of a song. This representation, here called a rhythmogram, can give much information about the rhythm and the evolution of the rhythm in time. An example of the rhythmogram for *Train to Barcelona* is shown in figure 2. The song seems to be a 4/4 with a tempo of 240 BPM, but in practice, the perceived beat is 120 BPM. In the first minute, it has an additional 8th beat, which is transformed into a 12th beat for the rest of the song, except a short period between 3 1/2 and 4 minutes, approximately.

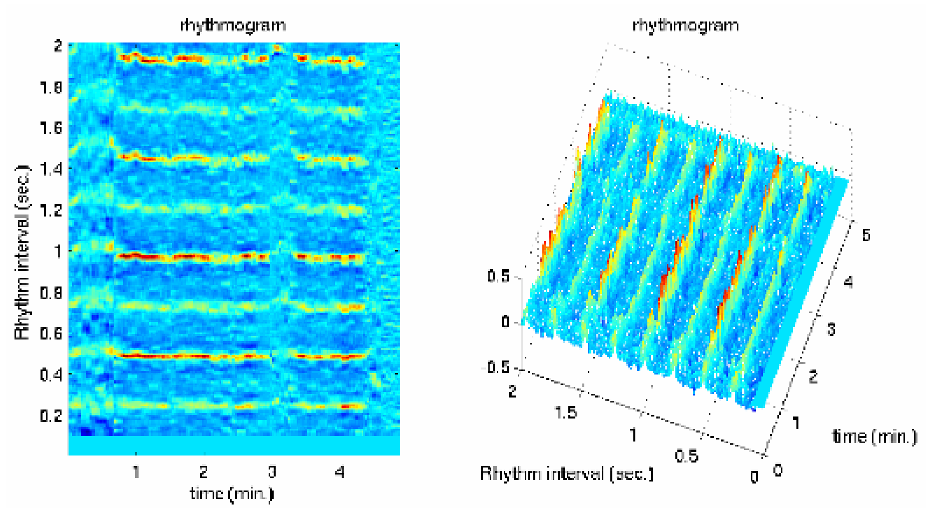


Figure 3. 2D and 3D rhythmogram for *I must be dreaming*.

Although the rhythmogram seems like a stable and robust representation, it can easily be shown that the robustness is, in part, caused by the gestalt behavior of the visual system. Indeed, if seen from another angle (in a 3D visualization), the rhythmogram reveals more movement, i.e. changes in relative strength of each beat in the measure, thus sometimes having different predominant beats in the measure. An example of such a 3D plot for I must be Dreaming, by Mink de Ville is shown in figure 3 (right). It is clear that it is not easy to segment the song according to a difference in rhythm. There seem to be an intro the first half minute, possibly repeated at around 3 minutes. Some change is taking place at around 1 1/2, 2 1/2 and 4 minutes, each time followed by a small change in tempo. As the song seemed to be played live, there is inherently an uncertainty in tempo, rhythm strength of each beat, and other timbre phenomena, which is all influencing to some degree on the rhythmogram.

3 Selfsimilarity

In order to get a better representation of the similarity of the song segments, a measure of self-similarity is used.

Several studies have used a measure of self-similarity [8] in automatic music analysis. Foote [10] used the dot product on MFCC sampled at a 100 Hz rate to visualize the self-similarity of different music excerpt. Later he introduced a checkerboard kernel correlation as a novelty measure [9] that identifies notes with small time lag, and structure with larger lags with good success. Bartsch and Wakefield [2] used the chroma-based representation (all FFT bins are put into one of 12 chromas) to calculate the cross-correlation and identify repeated segments, corresponding to the chorus, for audio thumbnailing. Peeters [21] calculated the self-similarity from the FFT on the energy output of an auditory filterbank.

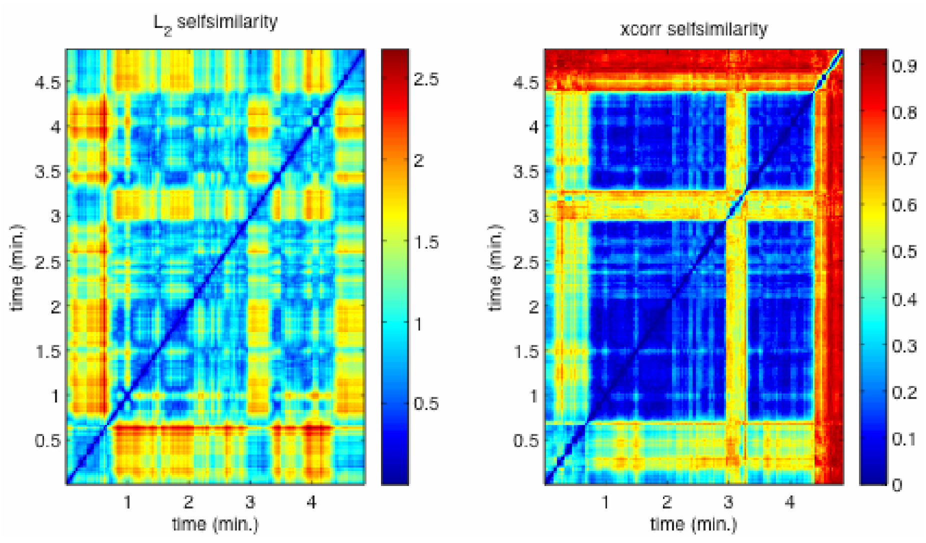


Figure 4. L_2 norm (left) and cross-correlation (right) self-similarity for *I must be dreaming*.

Generally, this measure of self-similarity is calculated directly on the feature(s), but in this case, an extra parameterization is introduced, the rhythmogram. The low sampling rate of the rhythmogram permits to calculate a rather small self-similarity matrix that is faster to calculate and easier to manipulate. In addition, as the rhythmogram seems to be close to a human perception of rhythm (cf. Desains decomposable theory of rhythm [6]), this intermediate step is also believed to make the self-similarity more directed towards rhythm than other features of the song, such as timbre. As the self-similarity should work, even if there is a drift in tempo, the cross-correlation self-similarity method is used, albeit it is significantly slower than the L_2 norm method. This has also been shown to minimize the L_2 norm between an audio feature and an expected a priori feature [14]. A comparison between the L_2 norm and the maximum of the cross-correlation method of *I must be dreaming* is shown in figure 4. The cross-correlation method (right in the figure) works best when there is a tempo drift in the song, which there is in most songs.

The self-similarity matrix can now be segmented, and the segments can furthermore be clustered. In this work, the song segmentation aspect will be detailed in the following section.

4 Causal Rhythm Grouping

The grouping, or segmenting, of a song, is the task of identifying segment boundaries that usually corresponds to boundaries humans would identify. The rhythm grouping indicates that orchestration and timbre is, as far as possible, omitted in the grouping, and the causal approach indicates that it is intended for possible real-time applications. In particular, the causal approach could permit the use of the identified segments in real-time composition, for instance using Murphys Pattern Play framework [20]. Another possible use is the identification of the 1st verse (or any particular rhythmic segment) in DJ software, such as Mixxx [1].

On related work, Bartsch and Wakefield [2], used chroma-based features to identify the repeated segment that corresponds to the chorus using cross-correlation. Foote [9] used cosine distance self-similarity and radially-symmetric Gaussian kernel correlation as a novelty measure that identifies notes for small lags and segments for large time lags. Dannenberg [4] made a proof-of-concept using pitch extraction and a matrix representation and melodic similarity algorithm on *Naimi* by John Coltrane. As a final step, the segments were clustered on three different songs. Peeters [21] converts the self-similarity to lag time and performs 2D structural filtering to identify segments.

The task is to find segments that consist of audio with similar rhythmic structure. As it is a causal approach, there is no knowledge about the rhythmogram ahead of the current time.

The approach chosen is to calculate the cross-correlation self-similarity matrix at a small lag time around the current time position only, and to calculate the novelty function [9] at these time lags. As the segments in the self-similarity matrix consist of squares around the diagonal, the boundaries of the squares can be identified by correlation the diagonal with a kernel that has the same shape. Foote gives the option of using either a binary checkerboard kernel, or to create a radially-symmetric Gaussian kernel. No significant difference was found between the two kernels in this work. An example of the novelty measure, calculated using the checkerboard kernel and three different kernel sizes, for *I must be dreaming* is show in figure 5.

It is clear that the small kernel sizes favors the note onsets (although only the relatively slow one, on the order of half the beat), whereas the large kernel sizes favors the structure in the song. In addition, the peaks are changing position between kernel sizes.

To identify the section boundaries, a method inspired from the scale-space community [16] in image processing is used. In this method, which, when used on images, is mimicking the way the images are blurred on a distance, the segment boundaries are found on heavily smoothed novelty measure, and the boundaries are then identified in the unsmoothed novelty measure.

The split-point time estimation is done on smoothed envelopes. The smoothing is performed by convoluting the novelty measure with a gaussian,

$$SNm_\sigma(t) = Nm * g_\sigma(t), \qquad g_\sigma(t) = \frac{1}{2\pi\sigma} e^{-\frac{t^2}{2\sigma^2}}. \tag{2}$$

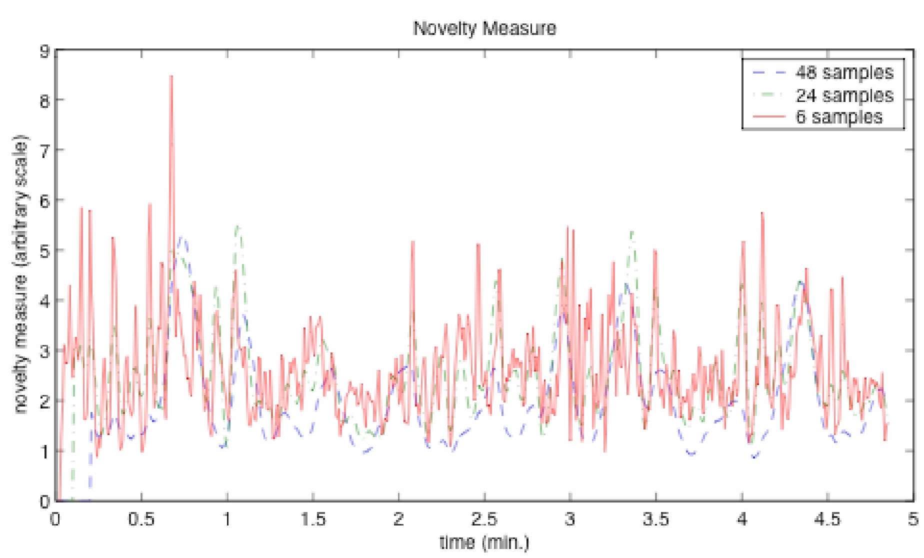


Figure 5. Novelty Measure for *I must be dreaming* and three different checkerboard kernel sizes.

The segment boundaries are now found by finding the zeros of the time derivative (with negative second derivative) of the smoothed novelty measure,

$$L_{t,\sigma}(t) == 0, L_{tt,\sigma}(t) < 0, \qquad L_{t,\sigma}(t) = \frac{\partial}{\partial t} SNm_{\sigma}(t), L_{tt,\sigma}(t) = \frac{\partial^2}{\partial t^2} SNm_{\sigma}(t). \tag{3}$$

The novelty measure is followed from the smoothed to the unsmoothed case in several steps by a method borrowed from the scale-space theory used, for edge detection, in image processing [16]. In case a peak is located near a slope, the slope influences the peak position when the novelty measure is smoothed. When the novelty function is less smoothed, it contains more noise, but the slope points correspond more to the unsmoothed case. It is thus necessary to follow the peak from the smoothed to the unsmoothed novelty measure, and to use enough smoothing steps so the slope points can be followed. An example of the smoothing steps, and the identified segment boundaries can be seen in figure 6.

Using an expert (the author) there is a certain resemblance between the intro, a segment at 3 to 3 1/2 minutes and the end segment. In addition, there are three segments consisting of verse-chorus at 0.5min to 1.5min, 1.5min to 2.5min, and 3.2min to 4.2min, the second of which the chorus lyrics is replaced with a guitar solo.

The automatic segment boundaries are found at 0, 0.2, 0.6, 1.5, 2.4, 3.3 and 4.4 minutes, where the zero and 4.4 minutes corresponds to the intro and end, the 0.6, 1.5 and 3.3 minutes corresponds to the verse chorus segments. The 0.2 minutes segment corresponds to the introduction of the vocal in the song. The second repetition of the intro theme was not found, but it seems that the automatic segmenting performs all in all almost as well as this expert. It is clear from the figure that there is much *novelty* in the song outside the found segment boundaries. More research is needed to assert whether these in fact correspond to perceptual boundaries or not. Another potential problem of the smoothing method is that it sometimes identifies a weak segment boundary in the middle of long segments, rather than a stronger boundary close to another boundary.

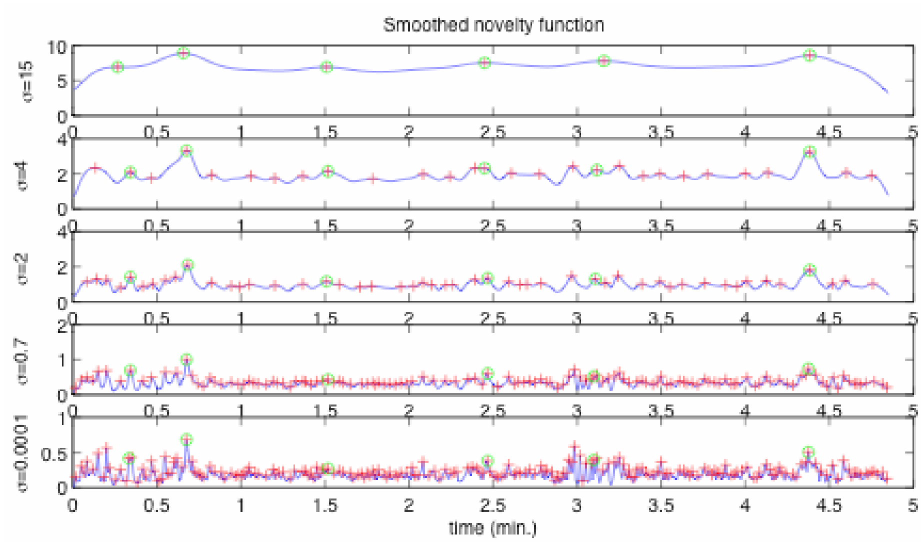


Figure 6. Example of the smoothed novelty function, peaks (‘+’), and the identified segment boundaries ('o') for *I must be dreaming*.

5 Evaluation

The segmentation steps are the feature extraction, the rhythmogram calculation, the self-similarity matrix calculation, the novelty measure, and the smoothing steps. The feature extraction is performed using an FFT in O($N \log_2(N)$) steps, the rhythmogram is calculated using an autocorrelation for each 8 seconds (800 steps), which can also be performed in $O(N \log_2 (N))$, the self-similarity matrix only needs to be calculated on the diagonal (4 new values for each time step), and novelty measure is smoothed in five steps. None of the last steps are very processor-intensive.

The segmentation has been performed on a small set (8) of rock and techno songs. Whereas the rock songs follow the intro, chorus, verse and break scheme well, the techno songs generally consists of long segments of music with no, or small evolutionary changes, and short consecutive segments with radical changes. The number of segments found is relatively stable for all songs, thus it seems that this method is useful for music summary, for instance. The automatic segment boundaries have been compared to human segmentation for the eight songs. First, it is obvious that some of the segment boundaries consist of vocal or other instrumental changes that are not found in the novelty measure. Around 10 % of the segment boundaries are not found, and the same amount has been misplaced by the unsmoothing peak following procedure. The smoothing makes it impossible to find short segments, which thus does not have to be prevented. Some of the misplaced peaks should possibly be found using help from some observations. For instance, it seems that some segment boundary peaks are preceded by a minima, i.e. before a change in rhythm, there is a short period with less than normal change. Another observation is that some segment boundaries are abrupt, but some consists of a gradual change where it is not clear (without counting beats and measures) where the boundary is.

The segmentation process was furthermore performed on a larger database of around three-hundred songs, consisting of child pop, pop, rock, noisy rock, world, classical, jazz, and possible other genres. A detailed analysis of the results has not been made, instead the performance of the segmentation system is evaluated using two statistics: the length of the segments, and the number of segments per song. These statistics are shown in figure 7.

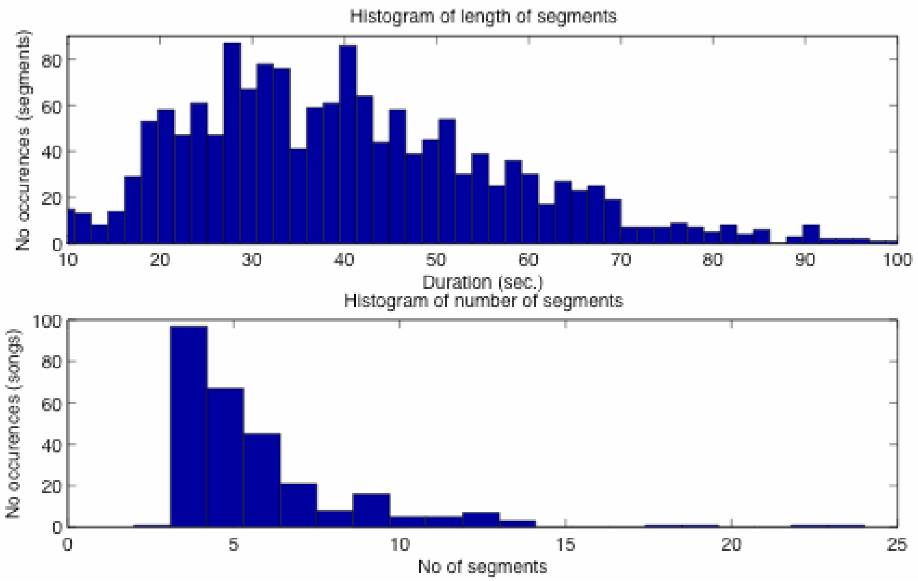


Figure 7. Statistics of the segmentation of a large number of songs. Length of segments (top), and number of segments per song (bottom).

It is clear that most songs are found to have a small number of segments. The extreme number of segments corresponds to four classical songs (Mozart and Schubert). No further analysis of the performance of the system in classical music has been made. An average duration of the segments of around 40 seconds seems reasonable, and although more analysis of the exact locations of the segments boundaries is necessary, it is concluded that in most respects the system is robust and reliable.

6 Conclusion

This paper has presented a complete system for the estimation of segments in music. The system is based on a hierarchical model, consisting of a feature extracting step, a rhythm model, and self-similarity step and finally a segment boundary identification step. The paper introduces a feature, the Perceptual Spectral Flux (PSF) that performs twice as good as a previously used feature. The rhythmogram is an intuitive model of the rhythm that permits an instant overview of the rhythmic content of a song. It is here used as a basis for the calculation of a similarity matrix [8]. In order to minimize the processing cost for the similarity matrix calculation, an efficient segment boundary method that only uses the diagonal of the self-similarity matrix has been devised, using the novelty measure [9] and a method inspired from the scale-space community in image processing [16].

The segmentation is intended to be used in real-time recomposition, in computer-assisted DJ software, and as an automatic summary generation tool.

All the steps have been verified with formal and informal methods. The audio feature (PSF) was found to be having a signal to noise ratio twice as good as the previously used feature, the High Frequency Content (HFC). The rhythmogram was shown to illustrate the rhythm pattern throughout a song. A 2D visualization was preferred, as it enabled following of rhythm patterns that were otherwise perceived as somewhat noisy in a 3D visualization. The self-similarity using cross-correlation was preferred, as the correlation permitted a better self-similarity measure in songs with a tempo drift. Finally, the segmentation was evaluated using a small database of rhythmic songs (rock and techno). Even though some of the verse-chorus segment boundaries could not be detected, as they consist mainly of lyric differences, most of the segments were identified correctly. An added benefit of this model is that it always identifies a suitable number of segments.

References

1. Andersen, T., H., Mixxx: Towards novel DJ interfaces, In proceedings of the New Interfaces for Musical expression, pp 30-35, 2003.
2. Bartsch, M. A. and Wakefield, G.H., To Catch a Chorus: Using Chroma-Based Representations For Audio Thumbnailing. *in* Proceedings of the Workshop on Applications of Signal Processing to Audio and Acoustics (CD), 2001, IEEE.
3. Casey, M.A.; Westner, W., Separation of Mixed Audio Sources by Independent Subspace Analysis, International Computer Music Conference (ICMC), pp. 154-161, August 2000
4. Dannenberg, R., ``Listening to `Naima': An Automated Structural Analysis of Music from Recorded Audio,'' In Proceedings of the 2002 International Computer Music Conference. San Francisco, pp. 28-34, 2002.
5. Deliege, I., Melen P., Cue abstraction in the representation of musical form *in* Perception and cognition of music, edited by Irène Deliège, John Sloboda. Hove, East Sussex, England. Psychology Press, pp. 387-412, 1997.
6. Desain P., A (de)composable theory of rhythm. Music Perception, 9(4), pp 439-454, 1992.
7. Dubnov, S., Assayag, G., El-Yaniv, R., Universal Classification Applied to Musical Sequences. Proc. of the International Computer Music Conference, Ann Arbour, Michigan, 1998.
8. Eckmann, J. P., Kamphorst, S. O., and Ruelle, D., Recurrence plots of dynamical systems, Europhys. Lett. 4, 973, 1987.

9. Foote, J., Automatic Audio Segmentation using a Measure of Audio Novelty. In Proceedings of IEEE International Conference on Multimedia and Expo, vol. I, pp. 452-455, July 30, 2000.
10. Foote, J., Visualizing Music and Audio using Self-Similarity. In Proceedings of ACM Multimedia, Orlando, Florida, pp. 77-80, 1999.
11. Goto M., and Muraoka, Y., A real-time beat tracking system for audio signals. In Proceedings of the International Computer Music Conference, pp. 171-174, 1995.
12. Goto, M., and Muraoka, Y., Real-time beat tracking for drumless audio signals: Chord change detection for musical decisions. Speech Communication, Vol 27. pp. 311-335, 1998.
13. Hermansky H., Perceptual linear predictive (PLP) analysis of speech, J. Acoust. Soc. Am., vol. 87, no. 4, pp. 1738-1752, Apr. 1990.
14. Jean Laroche., Efficient tempo and beat tracking in audio recordings, J. Audio Eng. Soc., 51(4), pp. 226-233, April 2003.
15. Jensen K. , T. H. Andersen. Real-time beat estimation using feature extraction. In Proceedings of the Computer Music Modeling and Retrieval Symposium, Lecture Notes in Computer Science. Springer Verlag, pp 13-22, 2003.
16. Lindeberg, T., “Edge detection and ridge detection with automatic scale selection”, CVAP Report, KTH, Stockholm, 1996.
17. Masri, P., and A. Bateman., Improved modelling of attack transient in music analysis-resynthesis. In Proceedings of the International Computer Music Conference, pages 100-104, Hong-Kong, 1996.
18. McAdams, S., Musical similarity and dynamic processing in musical context. Proceedings of the ISMA (CD), Mexico City, Mexico, 2002.
19. McAuley, J. D., Ayala, C., The effect of timbre on melody recognition by familiarity. Meeting of the A.S.A., Cancun, Mexico (abstract), 2002.
20. Murphy. D., Pattern play. In Alan Smaill, editor, Additional Proceedings of the 2nd International Conference on Music and Artificial Intelligence, On-line tech. report series of the University of Edinburgh, Division of Informatics, Edinburgh, Scotland, UK, September 2002. http://dream.dai.ed.ac.uk/group/smaill/icmai/b06.pdf.
21. Peeters, G., Deriving musical structures from signal analysis for music audio summary generation: sequence and state approach. *In* Computer Music Modeling and Retrieval (U. K. Wiil, editor). Lecture Notes in Computer Science, LNCS 2771, pp. 143-166, 2003.
22. Rauber, A., Pampalk, E., and Merkl D., Using Psycho-Acoustic Models and Self-Organizing Maps to Create a Hierarchical Structuring of Music by Musical Styles, Proceedings of the ISMIR, Paris, France. October 13-17, pp 71-80, 2002.
23. Scheirer, E., Tempo and Beat Analysis of Acoustic Musical Signals, Journal of the Acoustical Society of America, Vol. 103, No. 1, pp. 588-601, 1998.

Fugue Composition with Counterpoint Melody Generation Using Genetic Algorithms

Andres Garay Acevedo

Georgetown University – Washington DC, USA
ag66@georgetown.edu

Abstract. This paper presents the results of implementing and evaluating a genetic algorithm to assist in the task of automatic counterpoint generation. In particular, a fugue subject was used as an input for the system, while the generated counterpoint melody was to act as the countersubject. The genetic algorithm was tested with two different input melodies, and a basic set of rules for fitness evaluation. Within this domain, the results were satisfactory. Finally, the suitability of genetic algorithms for the task of rule-based melody generation, as well as possible future work and enhancements to the implemented system, are also reviewed and discussed.

1. Introduction

The task of composing music with the assistance of a well-defined algorithm has been a greatly debated over time. W. A. Mozart's dice game showed the feasibility of generating simple pieces with an aesthetical appeal by using nothing more than a set of musical motives and a pair of dice. Recently, the debate has focused on the use of computers to create music by obeying a strict set of rules defined by the composer [1].

Leaving the debate aside, one can see the task of composing music as that of applying formal rules over a starting melody until a satisfactory result is achieved. Mathematical relations, for example, are one of the important rules used during this process of incremental revision. The resulting piece is then a product of what Bruce Jacob defines as a "hard work composition" [9], a composition that doesn't come from a magical moment of inspiration but rather from hours of frustrating an arduous methodical work. Other authors use the term "compositional loop" [11].

One can then imagine a computer as a form of help in this process of *hard work* composing. The role of the composer isn't replaced by the computer, which just acts as an assistant that carries out part of the mentioned *arduous work*, thus leaving the composer with time to focus on other aspects of the composition. The task becomes then to identify the specific processes that are well suited to be carried by a computer.

In this paper we explain why counterpoint generation is such a process. Then we present the results of implementing a small-scale system that uses a genetic algorithm in order to find a suitable counterpoint melody that can be arranged in a basic fugue structure. Finally, some observations and possible future work are discussed.

2. The Problem

2.1 Motivation

Since William Schottstaedt proposed a system for automatic counterpoint melody generation [12], no other author has approached the problem by using a search algorithm. Instead, stochastic models have been primarily used instead for melody generation [13,8], while other authors use genetic algorithms to focus on higher aspects of the compositional loop [11].

Given this situation, it is valid to ask if genetic algorithms are a valid technique for solving the counterpoint melody search problem. Schottstaedt's system [12] used a best-first search algorithm to find a suitable solution for the problem. We, in turn, are interested in evaluating the suitability of genetic algorithms for this task. In particular, we are interested on counterpoint melody generation for the task of arranging a simple fugue. This is a two step process that is now presented.

2.2 Problem Definition

"For several hundred years composers have praised species counterpoint as one of the best ways to learn to write music" [12, pg. 199]. Some of the formal rules for a *good* counterpoint have even been stated since the eighteenth century; for example, J.J. Fux's "Gradus ad Parnassum", published in 1725, presents a compilation of these. Such a well studied process starts to suit the idea of computer assisted composition. In more general terms, the process of generating valuable musical phrases, themes or whole pieces can be stated as a computational task. For the particular case of counterpoint melody generation, the task can be formulated as a search problem: Given an initial musical phrase, find a suitable counterpoint melody from the space of all possible melodies.

However, this proves to be an NP-complete problem once the search space is formalized [12]. The search domain is essentially unlimited given the combinatorial possibilities of individual notes in time, rhythm, harmony and melody [3]. If there are 16 ways of *moving* from one note to the next, then a short musical phrase of 10 notes will generate a search space of 16^{10} possible solutions. Given this situation, it becomes necessary to impose some constraints to the search space (or to the decision making algorithm) in order to make a proposed solution feasible.

Once the problem of finding an appropriate counterpoint melody has been overcome, the compositional process can continue. Specifically, in this project the idea was to generate a set of valid counterpoint melodies and arrange them in order to create a Fugue. This encompasses the creation of an answer and two counter-subject melodies, given the initial subject of the fugue. The musical phrases are then arranged over time for three voices as shown in table 1.

The process of reviewing the counter-subjects and arranging them in the basic fugue structure can be performed both by a human composer as well as by a computer. If an algorithm is to be used, then it must incorporate higher-level concepts of musical arrangement and coherence; it must have some sort of aesthetic value. This constitutes a different problem than that of creating melodies; nonetheless, it is a task that has also been widely studied and described, as will be seen in the following section. For the purposes of this paper, we assume that a human performs this arrangement, based on his personal aesthetics.

Table 1. Basic fugue structure

VOICE	PART 1	PART 2	PART 3
Soprano	Subject	Counter-Subject 1	Counter-Subject 2
Alto		Answer	Counter-Subject 1
Bass			Subject

3. Literature Survey

As can be expected, researchers have focused on the specifics of music composition described here. While some study the task of melodic phrase generation, others do so for issues of musical arrangement. Nonetheless, systems that combine both tasks have been developed and proven to be of great utility.

3.1 Melody Generation

Most popular systems have focused on generating notes based on stochastic methods, where each note is produced based on its conditional probability, given its preceding notes [13,8]. Such probabilities could be modified so that a certain “style” is encouraged.

Less rigid systems that rely on nonlinear dynamical systems have also been implemented. These are described as systems of mathematical equations, and display a behavior found in a large number of systems in nature [1]. The way in which notes are calculated is through a process of iteration, where the solution to the equations is calculated and then fed back into the system as an input value for the following iteration. The generated solutions can be viewed as a set of n points in space; this set is known as an orbit. Of special interest among composers are chaotic orbits, which display semi-cyclical behavior. That is, the musical melodies creates seem to “wonder around” a musical motif but without repeating the same melody.

The systems mentioned above generate melodies from scratch; at most they have mathematical functions as input. Another approach to the problem consists of generating the resulting melodies based on a given musical phrase, which acts as an input. Probably the most successful implementation of such an approach is GenJam [2], a system that uses genetic algorithms in order to *improvise* over Jazz melodies.

For these systems, the quality of the produced musical phrase must not be judged by itself. Rather, the melodic content of a reference phrase is used for fair evaluation. Applications of these systems include melodic development, thematic bridging, and harmonization. Burton & Vladimirova [3] give a compilation of musical applications were genetic algorithms in particular have been used for melody generation.

The specific problem of counterpoint *solving*[1] has been formally stated as a search problem that aims to use the formal rules of counterpoint composition that have been defined over time [12]. Nonetheless, further development of the problem or alternative solutions have not been published lately.

3.2 Arrangement

Artificial Intelligence techniques have been widely used to model compositional tasks of *higher level*. Most of the systems seem to use a generator/modifier/selector approach that was first proposed by Lejaren Hiller during the 1950's [6]. In this approach, the output of a musical generator (such as the ones described above) is fed into the modifier and selector subsystem, which performs melodic development over the main theme and then arranges the melodies in a suitable way. Jacob calls these components composer, ear and arranger respectively [8].

The arranger subsystem can be implemented in different ways. It can be a complex rule-based system, such as David Cope's EMI [4], which produces a deterministic behavior. It can also be a looser system that, for example, just organizes the melodies so that they follow a given contour pattern.

Bruce Jacob's *arranger* [8] implements another approach that is widely used in algorithmic composition: interaction between the system and a human composer. Jacob's system uses a Genetic Algorithm that selects possible phrase candidates and then presents them to the user. Once the user selects the ones it considers most fit, the algorithm extracts their parameters and employs them in future iterations.

Moroni's Vox Populi [11] goes even further as this interaction occurs in real time. The human composer makes use of a graphical interface, where can adjust the parameters of the Genetic Algorithm as the composition is performed. Alternatively, he can draw two curves that represent which are represented as variables that guide the compositional process.

The particular problem of arranging a simple fugue is discussed by Milkie & Chestnut [10], although it can be said that their project consists of an instantiation of the more general framework developed by Lejaren Hiller. However, their use of genetic algorithms is limited only to the generation of the fugue's subject. In this respect, the work resented here can be considered both different as well as more general.

4. Implementation

A system that generates a counterpoint melody was implemented and evaluated. The system architecture is depicted in figure 1. This system uses a simple genetic algorithm (SGA) in order to generate a melodic phrase, given a referent melody. In particular, the input melody can be seen as the fugue subject, and the output melody as a counter-subject.

[1] Some of the literature refers to this task as species counterpoint.

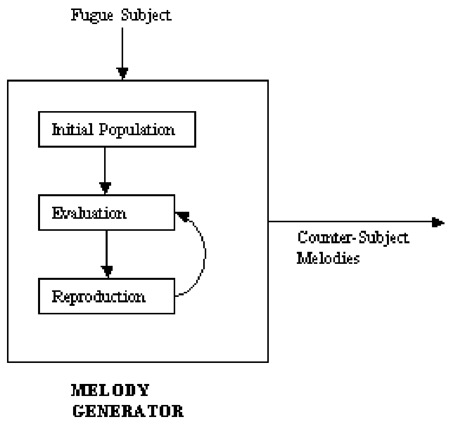


Fig. 1. System overview

When implementing a Genetic Algorithm (GA) based system it must be remembered that the system's success is dependent upon three characteristics: Search domain, fitness evaluation and input representation [3]. The search domain was specified in the previous section. In turn, how the latter elements were implemented is now discussed.

4.1 Melody and Note Representation

Melodies and notes were implemented as two different objects, using an object oriented programming language (OOP). Both representations are now discussed. First, however, it must be stated that the representation of a note is very similar to the one used by Biles [2] in GenJam, with the idea of a feature field borrowed from Milkie & Chestnut [10]. Similarly, the representation of a melody was borrowed from the latter.

Notes are characterized by three different attributes: pitch, length and feature. In the implementation, the *pitch* corresponds to an integer that represents the number of semitones between of the first note of the melody scale and the represented note. For example, if the note has a pitch value of 2 and the scale is C major, then the encoded note is a D. In particular, as notes were limited to a single octave for the experiment, the pitch value ranged between 0 and 11.

The *length* of each note was encoded as an integer, ranging from 0 to 3. A simplification was made here, as only 4 different lengths could be represented. This decision was made because we wanted melodies to have a quaver as the shortest note, given that this was the shortest note found on our input test set. Thus, the note durations encoded were: half a beat (0), one beat (1), two beats (2) and four beats (3).

Finally, the *feature* field was used to store extra information about a note. A value of zero was used to represent a simple note, a value of 1 to represent a silence or rest (thus eliminating the meaning of the pitch value), and a value of 3 to represent a hold.

Melodies, in turn, have five different parameters. The first one is the *key* of the melody. This was encoded using an integer value. In particular, we used the value 0 to represent a key of C major, the one used for all the melodies of our test. The second parameter is the *beat* of the melody. Here we also used an integer, and represented a beat of 4/4 with the value 0 for our test.

The melody *length*, in number of notes, was also stored as an integer value. The fourth parameter corresponds to the melody *fitness*, which was an attribute used specifically by the genetic algorithm. Finally, an array containing the actual notes is the last element that conforms a melody.

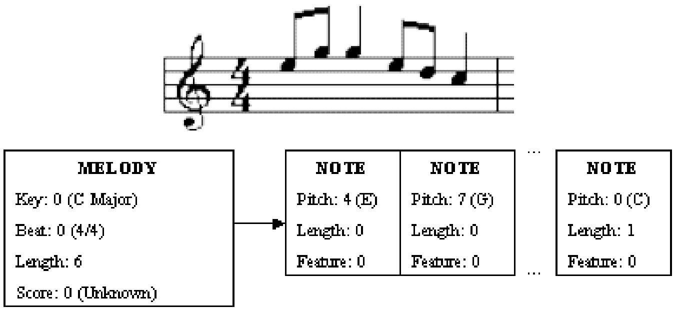


Fig. 2. A melody fragment and its representation

In order to better illustrate what has just been explained, a fragment of a melody and its representation are presented in figure 2. For each attribute, the store (encoded) value is shown after its name. The represented value is shown in parenthesis.

4.2 Fitness Evaluation and Crossover

The second element crucial to the performance of the genetic algorithm is the fitness evaluation. After carefully reviewing the literature, we decided to use the following function for the implemented system:

$$fitness = \sum_i weight_i * feature_i$$

where a set of features are evaluated and multiplied against a preset weight. This scheme is very similar to Schottstaedt's one of *prohibitions* and *penalties* [12], as some features were derived from his work. However, in order to adapt the rules for the genetic algorithm, melodies are rewarded when they display a certain feature, instead of being penalized for containing a *prohibition*. The set of selected features and its corresponding weights is presented in table 2.

Table 2. Selected features for fitness evaluation

Feature	Weight
Percentage of notes in the same key as the input melody	100
Same length in measures as the input melody	100
Melody ends in a consonant note	50
Melody begins with a consonant note	50
Percentage of intervals smaller than a third	100
Number of repetitions (higher than 2) of a note	-25

Once the fitness was calculated for each member of a generation, a single point crossover operator was used to create the offspring of a population. The crossover point was selected at random and corresponded to a particular measure at which the melodies are divided and then recombined to form two new melodic phrases. Figure 3 shows two sample melodies, and the resulting descendents when they are combined at the first measure. Note that if the value of zero is chosen as the crossover point, then the two parent melodies will be carried to the next generation.

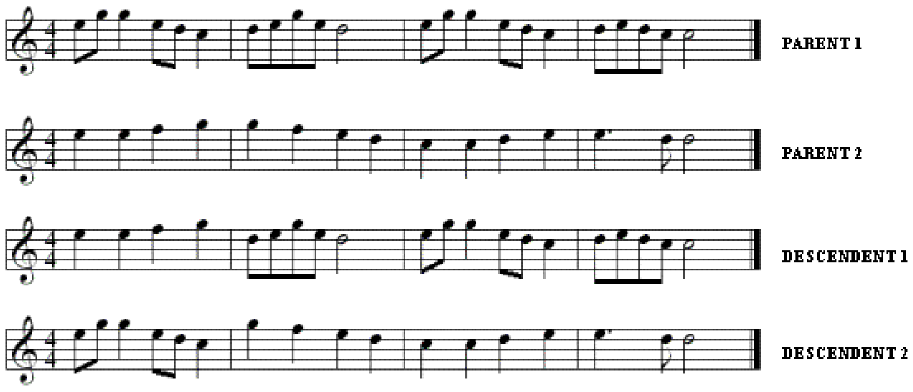


Fig. 3. Crossover at measure 1

By combining at a given measure instead that at a given note (i.e. at the third note), we could guarantee that the resulting melodies would have the desired length. This, in turn, guaranteed a minimum fitness on the newly generated population.

Finally, it is important for a genetic algorithm to have a mutation operator in order to assure that a suboptimal solution is found [5]. A mutation is no more than an evolution operation performed on an individual of a population. In particular for our case of musical phrases, mutation consists of a set of musical operations that are be performed on the original melody. The particular operations consist of transposition and inversion of a single note, randomly selected. These operators were selected to manipulate the content of the system and evolve the population in a musically meaningful way [3].

5. Measuring Performance

A test was carried to measure the quality of the melodies produced by the described system. In this section we describe how the test was performed, and then present the results found. Finally, these results are discussed.

5.1 Test Results

The genetic algorithm described in the previous section was run several times, using two different input melodies. Both of the input melodies were four measures long, with a beat of 4/4, and in the key of C major. This means that the output of the system was expected to present this same set of characteristics. Given the data representation selected, the fitness function and the crossover operator, all resulting melodies displayed these characteristics.

For the genetic algorithm, a population size of 50 individuals was used, and 100 generations were produced. These values were determined experimentally, and provided the best average fitness of the population.

Once the output melodies were collected, they were fed as the input of a MIDI sequencer. After this, they were played alongside a faded version of the input melody used for generating them. A sample melody, generated during the test, along with the input used for creating it are shown in figure 4.

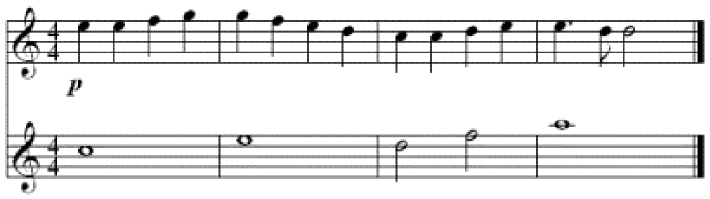


Fig. 4. Sample melody generated by the genetic algorithm

After collecting the results, the set of selected melodies was presented to an expert, who was then asked to grade them according to how well the generated melody could act as a fugue counter-subject for the input melody. That is, the generated melody was not evaluated by itself, but with respect to the system's input. A scale of 1 to 5 was used for this evaluation, were a value of 5 corresponded to the output melody being a good counter-subject for the fugue's main subject. A value of 1 represented no melodic relation between the two melodies.

The average grade given by the expert was 2.94. In addition, table 3 summarizes the percentage of melodies that were given each one of the five possible grades.

Table 3. Summary of results

GRADE	Percentage of Melodies
1	11.1%
2	16.7%
3	38.9%
4	33.3%
5	0%

Even though these figures might seem somewhat low at first sight, it must be stated that a grade of 5 corresponded to the melody being the work a musical expert (human or machine).

One final consideration that must be taken into account when evaluating the results of a melody generation system is that the generating algorithm must not converge upon a single solution [11]. Rather the solutions must display a set of recognizable characteristics without being a mere repletion of each other. There is not just one possible solution to the task of counterpoint generation.

The system tested displayed this desired behavior. In particular, no two melodies produced during the test are identical. This can be explained in part because the genetic algorithm was run for a fixed number of generations, and the algorithm doesn't converge upon a set of solutions at a fixed rate. That is, the average fitness of the final population is not necessarily the same for two repetitions of the same experiment.

This is a particularly interesting result, as this characteristic might not always holds true for counterpoint-solving algorithms based on other search techniques. Given this, genetic algorithms prove to be a viable technique for solving Schottstaedt's original formulation of the problem. Moreover, they might be a more suitable technique than other search algorithms, not only from a technical standpoint but also from a musical one. However, further testing and comparison must be done before making such a powerful claim.

5.2 Discussion

The results presented here are encouraging and promising. However, the system is far from being perfect; moreover, it's performance is not good enough to satisfy the needs of an average composer[2]. Further work and testing are both needed and encouraged. In anticipation of this, in this section we present a discussion of what we think are some of the necessary steps that must be taken in order to improve the results.

The first step needed in order to achieve better results consists of using a more elaborated fitness function for the genetic algorithm. By this we mean that a more complete set of features must be developed. Schottstaedt's set of rules (or penalties) provides a thorough compilation of the most important counterpoint rules, along with a measure of relative importance (their penalty). Other sources might be found in the musical literature.

This task shouldn't be very difficult, as it is highly localized; however it is important to state here another related problem, namely, that of data representation. The melody representation chosen is very useful and meaningful; nonetheless, the algorithm might benefit from a more detailed description.

Here a distinction must be made. The task of augmenting the set of features used by the algorithm relates directly to the problem at a higher level. On the other hand, that of using a more detailed data representation relates to the particular implementation. Nonetheless, with this distinction in mind we still find it useful to talk about the issue of data representation.

In particular, we think that some important data is lost when a melody is represented as a set of notes, grouped under a set of common characteristics. A more detailed description should make explicit the concepts of both measure and beat. Once these concepts are explicit, the set of operations that are performed on the population (such as crossover and mutation) will be much more efficient. Moreover,

[2] Species counterpoint exercises are not expected to produce great music.

implementing a more complete fitness evaluation function should become an easier task. This explains the importance of mentioning this particular implementation issue.

Our final recommendation pertains to the task of generating the initial population of the genetic algorithm. This task, we see, can be carried in two different ways. For our test system we decided to create the initial population of musical phrases using a random number generator, in accordance with most implementations of SGAs. In this way, the search space isn't constrained.

However, the initial population can also be created by using a stochastic note generator, such as the one proposed by Jacob [8]. This imposes a constraint on the search space; but that, in particular, might be a desired effect. By using a stochastic model for the creation of the initial population, a minimum level of fitness (and thus quality of the system) can be guaranteed. Moreover, some of the features that we used in our fitness evaluation can be eliminated. In particular, those that assess only to the individual being evaluated. If these features are removed, then the evaluation function will be comprised of a set of rules that refer only to the relation between the given individual and the referent melody. This can be then viewed as a cleaner evaluation, as it concentrates only on counterpoint specific features, rather than on those that relate both to the counterpoint task and the structural quality of the individual. However, it must be clear that what has just been described is a tradeoff between the clearness of the problem definition and modeling, and the completeness of the search space. It should be clear by this point that we decided to have a complete search space.

6. Conclusion

A basic algorithmic composition system has been proposed. The system finds a suitable set of melodies to be arranged in a fugue, given the fugue's main subject. Particular attention and effort has been put into the design of the melody generator, which uses a genetic algorithm in order to evolve generations of candidate solutions for the problem.

A test was carried to measure the quality of the results produced by the described system, and the results were discussed. In particular, we were very pleased with the results produced by one of the two test sets. Nonetheless, the overall results are also encouraging.

Future work and some specific issues were discussed. Some of these issues relate to the particular implementation of the system, while others relate to the problem specification. Both are considered important. Accordingly, future work is strongly encouraged.

Finally, it must be stated that the use of genetic algorithms has proven to be a viable technique for solving the problem of automatic counterpoint melody generation. Moreover, given some characteristics of the problem, such as the size of the search space and the existence of multiple solutions within this space, the use of a heuristic search method, and genetic algorithms in particular, seems to be a well suited mechanism for achieving a satisfactory solution.

References

1. Alpern, A. 1995. "Techniques for Algorithmic Composition of Music." On the web: http://hamp.hampshire.edu/~adaF92/algocomp/algocomp95.html
2. Biles, J. 1994. "GenJam: A Genetic Algorithm for Generating Jazz Solos." *Proceedings of the 1994 International Computer Music Conference*. San Francisco: International Computer Music Association.
3. Burton, A., and Vladimirova, T. 1999. "Generation of Musical Sequences with Genetic Techniques." *Computer Music Journal* 23(4): 59-73.
4. Cope, D. 1992. "Computer Modeling of Musical Intelligence in EMI." *Computer Music Journal* 16(2): 69-83.
5. Goldberg, D. 2000. "The Design of Innovation: Lessons from Genetic Algorithms, Lessons for the Real World." *Technological Forecasting and Social Change* 64: 7-12.
6. Hiller, L. 1981. "Composing with computers: A progress report." *Computer Music Journal* 5(4): 7-21.
7. Horner, A. and Goldberg, D. 1991. "Genetic Algorithms and Computer-Assisted Music Composition." *Proceedings of the 1991 International Conference on Genetic Algorithms*. San Mateo: International Society for Genetic Algorithms.
8. Jacob, B. 1995. "Composing with Genetic Algorithms." *Proceedings of the 1995 International Computer Music Conference*. San Francisco: International Computer Music Association.
9. Jacob, B. 1996. "Algorithmic Composition as a Model of Creativity." Organised Sound 1(3).
10. Milkie, E., and Chestnut J. 2001. "Fugue Generation with Genetic Algorithms". On the web: http://www.cs.cornell.edu/boom/2001/Milkie
11. Moroni, A. et al. 2000. "Vox Populi: An Interactive Evolutionary System for Algorithmic Music Composition." *Leonardo Music Journal* 10: 49-54.
12. Schottstaedt, W. 1989. "Automatic Counterpoint." *Current Directions in Computer Music*, pp. 199-213. Cambridge, Massachusetts: MIT Press.
13. Supper, M. 2001. "A Few Remarks on Algorithmic Composition." *Computer Music Journal* 25(1): 48-53.
14. Temperley D., and Sleator D. 1999. "Modeling Meter and Harmony: A Preference-Rule Approach." *Computer Music Journal* 23(1): 10-27.

Harmonizations of Time with Non Periodic Ordered Structures in Discrete Geometry and Astronomy

Juan García Escudero

Facultad de Ciencias, Universidad de Oviedo,
33007 Oviedo, Spain
jjge@pinon.ccu.uniovi.es

Abstract. Non periodic ordered tilings can be used for the generation of discrete structures in the time and frequency domains. The Fourier spectrum of impulse distributions ordered according with certain types of aperiodic ordered temporal sequences described by Lindenmayer systems shows a discrete part . In order to appy these ideas the main tools belong to discrete geometry and number theory. These techniques provide a connection between rhythms and harmonic fields which may have a natural phenomena basis when observational data of certain types of variable stars are analyzed. The pulsation of some semiregular and delta scuti stars is reflected in their light curves which can be modelled by means of sinusoidal sequences related with the golden number.

...Comment le monde doit se replier sur lui-même,se redoubler, se réfléchir ou s´ enchaîner pour que les choses puissent se ressembler... (Michel Foucault)

1 Introduction

One of the last trends in XX century is spectralism which uses harmonies derived from overtone series of natural sounds. The primary source of the techniques considered in this paper is not the spectrum of sounds but of temporal sequences. The main motivation lies in a series of recent works, where I have explored relations between aperiodic but ordered temporal sequences, harmonic fields arising from the Fourier analysis of such sequences and of instrumental sounds, and sounds synthesized starting from the dynamic spectra appearing in the analysis.

The basic structures are related with quasicrystals which are a new type of alloys discovered twenty years ago exhibiting a type of order different from either crystals or amorphous materials. The structures, which lie somewhere between periodicity and randomness, have been described by the author with the help of deterministic and stochastic Lindenmayer systems. In 2D and 3D the recursive structure of the geometries is represented in terms of bracketed word sequences [4].

Time is structured with 1D geometric sequences generating aperiodic ordered rhythms. The use of number theory shows that some of them are transparent in the sense that their Fourier spectra have a discrete component and provide the pairs intensity-pitch. The resulting harmony is an extension of the classical harmony which is based on the spectra of periodic rhythms. Due to the lack of translational symmetry, the temporal spectra are always inharmonic. The central frequencies in additive synthesis of filtered noises have been obtained from the basic temporal and instrumental spectra. Their amplitudes evolve according to successive iterations previous to the amplitudes stabilization. The bandwidths in [5] have been introduced according to perceptual criteria, but 1D non periodic ordered tilings can show also continuous components in their Fourier transform . In other works spectra of instrumental or vocal sounds are part of autonomous harmonic fields or in interaction with those corresponding to the spectra arising in the temporal structures. The scaling factors of the aperiodic ordered 1D sequences are algebraic integers : roots of monic (the leading coefficient is equal to 1) polynomials with integer coefficients. Certain connections with observational data in astronomy may also be established and the time can be structured in a natural phenomena basis.

2 L-systems for Quasiperiodic Tilings and Their Time and Frequency Representations

An approach to continuity can be obtained in computer music compositions due to the high accuracy of the medium. It has been also the basis of instrumental works by authors like Brian Ferneyhough: the pitches, dynamics and rhythms are chosen from a parametric space as continuous as possible from a perceptual point of view. Local realizations are, except in some cases like glissandi or dynamic changes, essentialy discrete. Several authors have looked in the past for discrete time structures that would be related to organizations of pitch structures [2]. By decreasing the duration of a series of impulses periodicaly distributed in time, Stockhausen pointed out how the same basic process is behind our perception of duration and pitch [16] . Longer durations define temporal scales related with the form and its articulations.

A possible way to structure the time is by means of non periodic ordered sequences which can be described algebraically in terms of Lindenmayer systems. A 0L-system [15] is a triple $G = \{\Sigma, r, \Omega\}$ where Σ is an alphabet, r is a finite substitution on Σ into the set of subsets of Σ^*, and Ω is the axiom. G is called a D0L-system if $\#(r(x)) = 1$,for every $x \in \Sigma$.

Consider the alphabet $\Sigma = \{a, b\}$, the production rules:

$$a \longmapsto abaa, \; b \longmapsto aab \tag{1}$$

and the axiom a. The language consists in the words $a, abaa,$ $abaaaababaaabaa, ...$ If we associate to a and b temporal segments (rhythmic units) of lengths l_a, l_b with $l_a/l_b = (\alpha - 1)/2$, where α is the highest root of the polynomial $x^2 - 4x + 1$, we get a selfsimilar sequence with scaling factor α. Although α is not rational, rational approximants can be obtained for the basic rhythmic units by using the recursion relations

$$a_{n+1} = 3a_n + b_n,\, b_{n+1} = 2a_n + b_n \tag{2}$$

Impulse distributions are now placed on the points separating consecutive cells. This is represented by the function

$$\rho(t) = \textstyle\sum_k \delta(t - t_k) \tag{3}$$

When the points t_k belong to a periodic lattice then the spectrum obtained is the familiar overtone series, with all the frequencies having the same intensity. In our case the duration t_n of the temporal sequence after iterating n times the production rules to the axiom a satisfies

$$t_{n+1} = 4t_n - t_{n-1} \tag{4}$$

and initial conditions $t_0 = l_a$, $t_1 = 3l_a + l_b$. The frequencies forming the discrete part of the Fourier transform of impulses distributions following the sequence defined by eq.1 have the form :

$$\omega = p\nu_1 + q\nu_2 \tag{5}$$

where p and q are arbitrary integers and $\nu_1 = (\alpha - 1)/(\alpha + 1)$, $\nu_2 = 2/(\alpha + 1)$. If $c(n)$ denotes the number of impulses the Fourier transform of $\rho(t)$ is

$$\lim_{n \mapsto \infty} \frac{1}{c(n)} \sum_n \exp(2\pi i t_n \omega) \tag{6}$$

and the recursion relations for the Fourier amplitudes are

$$\begin{aligned} &F_{n+1} = F_n(1 + exp(it_n\omega) + exp(2it_n\omega) + exp(i(3t_n - t_{n-1})\omega))- \\ &F_{n-1}exp(i(3t_n - t_{n-1})\omega) \end{aligned} \tag{7}$$

Four iterations are needed in order to stabilize the quotient between the amplitudes and the number of impulses. The static spectrum is represented in Fig.1. Although the spectrum is a dense set, above a certain amplitude thereshold the number of frequencies which are lower than a given one is finite. The static spectra contains pitches separated in the frequency space by distances of two lengths. The first pitches of a spectrum with this property are:
$\{1, 0, 58, f, -0.95\}, \{2, 0, 115, pp, -1.7\}, \{2, 1, 158, ff, 0.7\}$,
$\{3, 1, 215, ff, -0.3\}, \{4, 1, 273, mf, -1.2\}$,
$\{4, 2, 315, mf, 1.3\}, \{5, 2, 373, ff, 0.4\}, \{6, 2, 431, ff, -0.5\}$,
where in each quadruple we use the notation:
$\{p, q, frequency(Hz), intensity, phase\}$, and p, q are the integers in eq.5.

For generic substitutional sequences the Fourier spectrum is a sum of discrete, continuous and singularly continuous (its derivative vanishes almost everywhere)

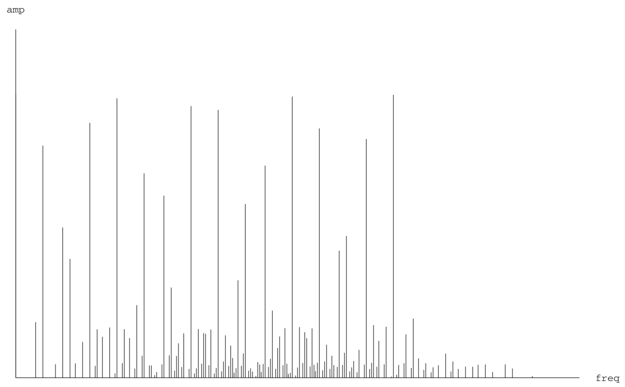


Fig. 1. Plot of the normalized Fourier amplitudes for the ab-system.

components. When the scaling factor is a Pisot-Vijayaraghavan number, namely, an algebraic integer greater than one with all its conjugates (the remaining roots of its minimal polynomial) strictly less than one in modulus, then there is, up to possible extinctions, a discrete component in the Fourier transform [1]. This is the case for the sequences considered above. In [7] continuous approaches (which does not mean a continuos component) to the spectrum have been given for the system with production rules $r : \{a^0 \longmapsto b^1, a^1 \longmapsto b^0, b^0 \longmapsto a^1c^0, b^1 \longmapsto c^1a^0, c^0 \longmapsto db^0, c^1 \longmapsto b^1d, d \longmapsto c^0c^1\}$, with $l_a = sin[\pi/8]; l_b = sin[2\pi/8]; l_c = sin[3\pi/8]; l_d = sin[4\pi/8]$. In fact the substitution gives two types of sequences depending on which iteration level one considers. It is possible to construct substitution rules related with one level $r : \{a \longmapsto ac, c \longmapsto ccac\}$ such that the inflation factor is a Pisot number and then, up to extinctions, there is a discrete component. A substitution rule with a non Pisot scaling factor and hence no discrete component in the Fourier transform is $r : \{a \longmapsto acca, c \longmapsto ccaccacc\}$. Figs.2,3 show a plot of the normalized amplitudes corresponding to two successive words with lengths 56 and 384. A progressive but irregular emergence of sharp peaks with well defined positions can be seen as in the non-Pisot structure studied in [11].

The 1D tilings discussed are the basis for the construction of 2D aperiodic geometries which may formalize harmonic fields progressions. In [9] several planar tilings with eight-fold symmetry have been introduced. They are constructed by means of a substitution process and the complexity is obtained by iteration of simple rules as in other scientific fields. The method can be applied because each prototile can be subdivided into smaller rotated copies of themselves and the other prototiles. A biological motivation is behind tiling growth: many cell divisions in multicellular organisms occur at the same time. The recursive structure of the figures is captured in terms of word sequences in bracketed Lindenmayer systems where productions are applied in parallel and simultaneously replace all the letters in a given word.

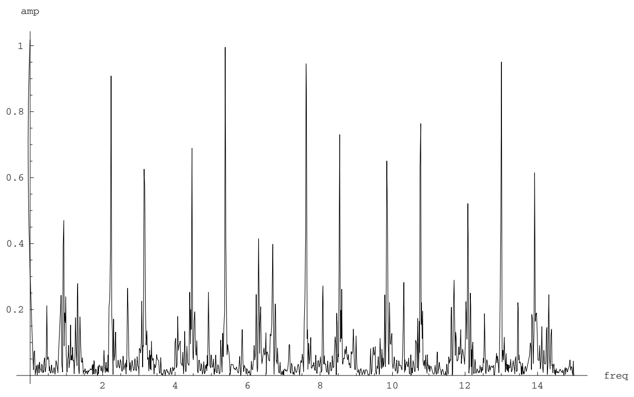


Fig. 2. Plot of the normalized Fourier amplitudes for the non-Pisot ac-system with 56 impulses.

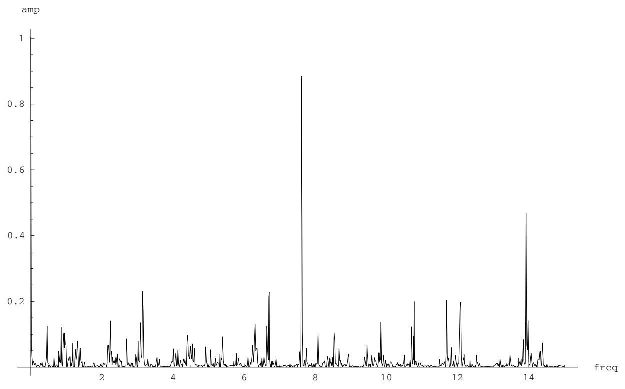


Fig. 3. Plot of the normalized Fourier amplitudes for 384 impulses.

For the substitution tilings derived in [9] the prototile edges A, B, C, D, E, F have respective lengths $2c_1, 1, 2c_2, 2c_1c_2, 2c_2c_3, 2c_3$, with $c_\nu = \cos(\nu\pi/8)$. They are selfsimilar with the Pisot-Vijayaraghavan inflation factor $1 + 2c_2$, highest root of $x^2 - 2x - 1$. The letters $\alpha_m, \beta_m, \gamma_m, \delta_m$ represent the prototiles $T(B, B, A), T(B, B, C), T(D, E, C), T(B, F, B)$ respectively, where $T(X, Y, Z)$ is a triangular tile with edges X, Y, Z placed anticlockwise. Letters of type t and $\widetilde{t}$ represent mirror images. The prototiles of type $\widetilde{t}$ are marked with a symbol o in Fig.4. The alphabet is

$$\Sigma = \{\alpha_m, \beta_m, \gamma_m, \delta_m, \widetilde{\alpha}_m, \widetilde{\beta}_m, \widetilde{\gamma}_m, \widetilde{\delta}_m, (,)\} \tag{8}$$

with $m \in Z_{16}$ (Z_m denotes the set of integers module m). The tile $T(X, Y, Z)$ with the edge Z placed on the positive x-axis corresponds to t_1 and the mirror

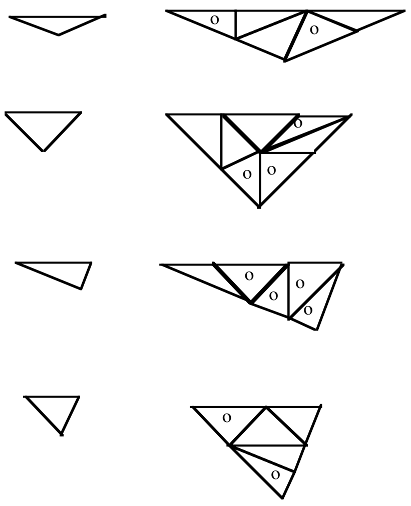


Fig. 4. The substitution rules for the prototiles.

reflection through an axis perpendicular to Z gives $\widetilde{t}_1$. The oriented tiles t_i and $\widetilde{t}_i$ are obtained by a rotation of $\pi(i-1)/8$ through the left most vertex.

Every element belonging to the alphabet representing a tile can be used as axiom. The set of production rules is (Fig.4):

$$\begin{aligned}
&\alpha_m \longmapsto (\widetilde{\delta}_{m+7}\delta_{m+9}\beta_{m+1}\widetilde{\beta}_{m+9}\alpha_m)\\
&\beta_m \longmapsto (\beta_{m+6}\delta_{m+14}\beta_m\widetilde{\alpha}_{m+9}\alpha_{m+1}\widetilde{\beta}_{m+10}\widetilde{\delta}_{m+6})\\
&\gamma_m \longmapsto (\alpha_{m+7}\widetilde{\beta}_m\beta_{m+8}\gamma_m\delta_m)\\
&\delta_m \longmapsto (\alpha_{m+7}\widetilde{\beta}_m\widetilde{\delta}_{m+12}\widetilde{\beta}_{m+10}\widetilde{\gamma}_{m+2})\\
&) \longmapsto)\\
&(\longmapsto (
\end{aligned} \tag{9}$$

In the word $((\widetilde{\beta}_m\beta_{m+8}\delta_m)(\gamma_m\widetilde{\gamma}_{m+6}))$, if two letters follow one another inside a bracket, the corresponding oriented triangles are glued face to face in a unique way. The oriented trapezoid $(\widetilde{\beta}_m\beta_{m+8}\delta_m)$ is then glued also face to face with the oriented triangle $(\gamma_m\widetilde{\gamma}_{m+6})$ disregarding their internal composition and again the prescription is unique. A part of the infinite planar tiling can be seen in Fig.5.

This type of word sequences , which were introduced for the description of Penrose patterns [4], have been used in the formal constructions since [3]. The prototiles may have also a correspondence in terms of elementary harmonic fields [6] . They can be derived from the prototile edges, their inflated versions or the spectra associated with the edge substitution rules (see eq.10). Local periodicities can give , in some cases, chords close to those classified by the classical harmony and the words describing the tiling growth produce a hierarchy of chord sequences.

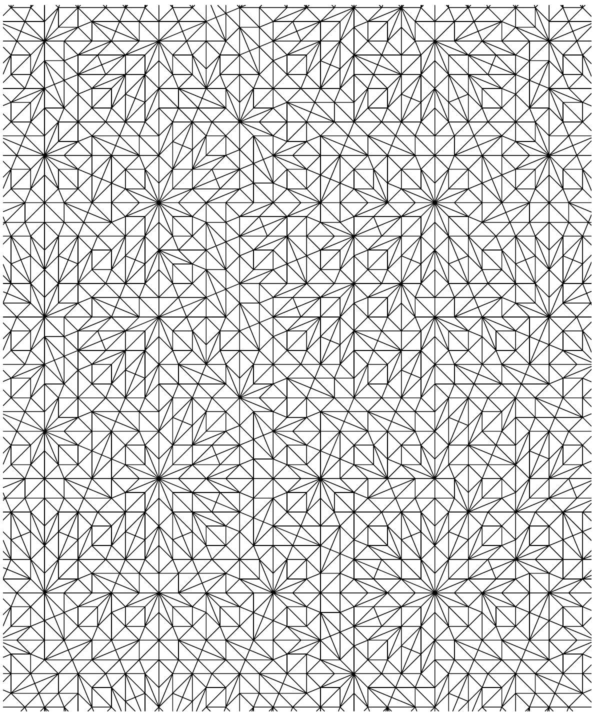


Fig. 5. A fragment of the infinite octagonal tiling with four prototiles.

Different species of planar patterns ,related with number twelve, have been generated in [8] . The prototile edges are $A = s_1, B = s_2, C = s_3, D = s_4, E = s_5, F = s_6$. where $s_l = \sin(l\pi/12)$. An arrow is placed on all the edges except F. For a given prototile their edges are labelled with $0, 1$ depending on whether the arrow orientation with respect to the prototile interior is anticlockwise or clockwise respectively. The substitution rules $X \longmapsto \phi[X]$ are :

$$\begin{array}{ll} \phi[A^0] = E^1 & \phi[B^0] = D^0F \\ \phi[C^0] = E^1E^0C^1 & \phi[D^0] = D^0FD^1B^0 \\ \phi[E^0] = A^1C^0E^1E^0C^1 & \phi[F] = B^1D^0FD^1B^0 \end{array} \qquad (10)$$

and $\phi(L^i) = Mir(t(\phi(L^{i+1})))$ where Mir denotes the mirror reflection of a word $i \in Z_2$ and the map t increases in one unit the index.

In what follows we use the rhythmic notation of Patchwork-Open Music. The list ((4 (1 1 1 1) 3 (1 1 1)) contains two sub-lists each one representing a bar, the number before each sub-list denotes the number of pulsations and negative

numbers indicate rests. For instance 4(-2 1 1) denotes a half-note rest followed by two quarter-notes . In Yod , a work for six percussionists and computer, the prototile edges substitution rules given in eq.10 play a major role. It is possible to define also rational approximants for the temporal segments s_l along the lines of eq.2. The last section of Yod begins with the following rhythmic pattern in the first bar:
E :(4 (6 (1 -2 1 -2) 6 (-1 1 1 2 1) 6 (-3 1 2) 6 (-1 1 -2 1 -1)) ,
C: (4 (11 (1 -3 1 -3 1 -2) 11 (1 -2 1 -3 1 -3) 11 (11 -2 1 -3 1 -2) 11 (-1 1 -2 1 1 -3 1 -1)) ,
F : (4 (11 (1 -1 1 -3 1 -4) 11 (1 -3 1 -3 1 -2) 11 (-2 1 -3 1 -1 1 -2) 11 (-2 1 -3 1 -1 1 -2)) ,
A: (4 (11 (1 -3 1 -2 1 -2 1) 11 (-3 1 -2 1 1 -3) 11 (1 -2 1 -3 1 -2 1) 11 (-2 1 -3 1 -3 1)) ,
D : (4 (16 (1 -1 1 -2 1 -3 1 -2 1 -1 1 -1) 16 (-1 1 -3 1 -2 1 -1 1 -2 1 -2) 16 (-1 1 -2 1 -3 1 -2 1 -1 1 -1 1) 16 (-2 1 -2 1 -1 1 -1 1 -2 1 -3)) ,
B: (4 (16 (1 -8 1 -6) 16 (-1 1 -7 1 -6) 16 (-1 1 -4 1 -8 1) 16 (-7 1 -8)).

The letter preceding each sequence is the axiom that generates it. Observe that there are two independent subsets $\{A, E, C\}, \{B, D, F\}$ in eq.6 which transform into themselves and therefore every sequence has only three rhythmic units. They are constructed in such a way that the only whole coincidences are at the begining and at the end of the section.

3 Pulsations of Variable Stars and Substitutional Sequences

In addition to the aperiodic ordered structures which have a mathematical motivation, spectra of natural phenomena in astronomy can be used as part of the pitch and rhythmic organizations. Stellar pulsations may be irregular, semiregular or periodic. In this section I consider stars with light curve variations related with the structures previously discussed. Semiregular stars like UW Herculis and some Delta Scuti stars have a multimode pulsation with a light curve that can be modelled by concatenation of sinusoidal fragments following word sequences in formal grammars. The best known example of a selfsimilar quasiperiodic sequence is the Fibonacci sequence. It is related with the ubiquitous golden number . Lindenmayer systems can be used in order to describe the Fibonacci sequences . The alphabet is $\{L, S\}$, the production rules $r : \{L \longmapsto LS, S \longmapsto L\}$ and the axiom L. The sequence consists in the words $L, LS, LSL, LSLLS, ...$ A 1D quasiperiodic geometric structure can be obtained if L and S represent two segments with a ratio equals the golden number τ. The sequence is deterministic because only one word is allowed with a given length. The Fibonacci numbers $F(n)$ can be defined with the help of the recurrence relation $F(n+2) = F(n+1) + F(n)$,$F(0) = F(1) = 1$. By iterating this relation we get the sequence 1,1,2,3,5,8,13,21... The quotient of two succesive Fibonacci numbers approaches the golden number when n increases. The following rhythmic sequence corresponds to a subdivision of a segment $L = 34$:

(7 (5 (1 -4) 5 (-5) 5 (-5) 5 (-5) 5 (-1 1 -3) 5 (-5) 5(-5))) $\mapsto$ (7 (5 (1 -4) 5 (-5) 5 (-3 1 -1) 5 (-5) 5 (-1 1 -3) 5 (-5) 5 (-5))) $\mapsto$ (7 (5 (1 -4) 5 (-3 1 -1) 5 (-3 1 -1) 5 (-5) 5 (-1 1 -3) 5 (-4 1) 5 (-5))) $\mapsto$ (7 (5 (1 -4) 5 (1 -2 1 -1) 5 (-3 1 -1) 5 (-3 1 -1) 5 (-1 1 -3) 5 (-1 1 -2 1) 5 (-5))) $\mapsto$ (7 (5 (1 -2 1 -1) 5 (1 -2 1 -1) 5 (-1 1 -1 1 -1) 5 (-1 1 -1 1 -1) 5 (-1 1 -2 1) 5 (-1 1 -2 1) 5 (-2 1 -2)))

The sequences do not have translational symmetry: they can not be derived, independently of its length, by translation of a unique subsequence. They show a certain amount of redundancy because, when the infinite word is considered, any block of consecutive symbols can be found also at some place in the word. These properties have their geometric correspondences also in the infinite planar tilings discussed (Fig.5): they can not be generated by translation of a single geometric configuration and every patch of tiles occurs at some place in the tiling. If delta functions are ordered according with this rhythmic sequences then it is possible to get recursion relations along the lines of eq.. The spectrum has a discrete component which is also a dense set. The order that lies in the temporal sequence is reflected in the frequency distribution and we can extract a set of partials with distances of two lengths in a golden ratio, although the words obtained in the frequency space do not belong to the grammar generating the rhythmic structures.

A different approach is suggested by a method recently introduced in astronomy [10]. The light curve of the Delta Scuti star V346 Orionis can be modelled with a curve constructed by concatenation of two sinusoidal fragments with two lengths in a golden ratio. For the observations taken on the night of 4 November 2001 at the 1.52 telescope in Loiano (Italy) [13], the temporal segments have been chosen by matching with the empirical dataset. In Fig.6 are plotted differential stellar magnitudes of V346 Ori relative to the star HD 35351 (see [13]). In the abscise the modified Julian days correspond to MHJD=(HJD-2452218)x10^5 . For the generation of the artificial light curve, represented as a continuous curve in Fig.6 , the axiom is SL and the derivation corresponds to $r^3(SL) = LSLLSLLS$.

The Fourier spectrum of an artificial light curve with two sinusoidal fragments with lengths $L = F(10) = 89$ and $S = F(9) = 55$ units following a Fibonacci sequence can be seen in Fig.7 The analysis corresponds to the word r^{11} with 144 letters (compare [10] for the analysis corresponding to the word with 89 letters). The periods corresponding to the peaks with highest amplitudes in Fig.7 have 123 and 76 units, a factor of $3 - \tau$ in relation with the durations of the sinusoidal fragments. The presence of sharp peaks is probably due to the fact that the golden number is a Pisot-Vijayaraghavan number (in this case the minimal polynomial is$x^2 - x - 1$) . In [12] it is shown that any Pisot substitution dynamical system on two symbols has pure discrete spectrum. The frequencies of the Fourier spectrum of Fibonacci sequences belong to a dense set which consists of the linear combinations with integer coefficients of two fundamental frequencies: $f = m_1\omega_1 + m_2\omega_2$, where ω_1 and ω_2 are in a golden ratio and m_1, m_2 are integers. The spectrum stabilizes when the iteration level is increased. The sharp peaks appear in the analysis when the word length is long enough. For

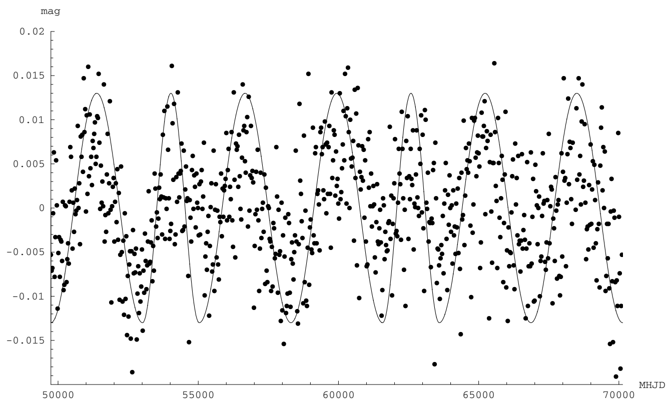


Fig. 6. V346 Orionis light curve and the artificial curve corresponding to the sequence containing the first seven letters of the word $r^3(SL)$.

shorter words the peaks appear with a certain bandwidth and this fact can be used in order to get dynamic spectra for direct synthesis of sound by appropriate scaling the basic units.

The semiregular star UW Herculis can be studied with the help of models based on stochastic sinusoidal sequences. A stochastic 0L-system is a 4-tuple $G = \{\Sigma, P, \Omega, \pi\}$ where P is a set of productions r_i and $\pi : P \longmapsto (0, 1]$ is a probability distribution. We define a Fibonacci stochastic L-system with $P = \{r_1, r_2\}$ and $r_1 : \{L \longmapsto LS, S \longmapsto L\}, r_2 : \{L \longmapsto SL, S \longmapsto L\}$. The Fourier spectrum of artificial data generated by following the word sequences of this system contains peaks with bandwidths. When the amplitudes are increased at certain positions according with the observations, some of the pitches of the spectrum with significant amplitudes are (the frequency in Hz and the intensity are specified in each pair)
$\{53, f\}, \{124, mf\}, \{178, f\}, \{231, f\}, \{267, mf\}, \{320, mf\}$,
$\{373, f\}, \{427, mf\}, \{498, mf\}, \{551, mf\}, \{605, ff\}, \{960, mp\}$,
which corresponds to a certain period of time. It has been part of the structural models considered in Yod. Dynamic spectra can be generated by including different periods of time .

4 Concluding Remarks

In this paper specific techniques with a basis in number theory, geometry and astronomy have been discussed in an attempt to implement, in a different context, formalized mechanisms founded on the mathematics of aperiodic ordered systems. A common intention is the computation of static and dynamic spectra of temporal sequences having a mathematical or physical motivation. From the compositional point of view the emphasis is shifted from the spectrum of

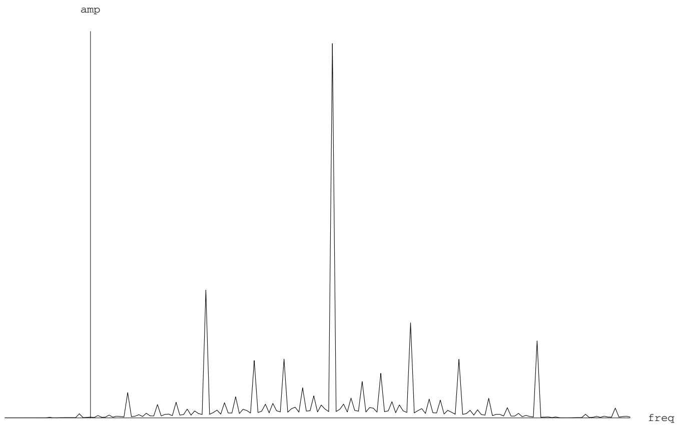


Fig. 7. Normalized Fourier amplitudes for the curve with sinusoidal fragments following a Fibonacci sequence with 144 symbols.

sounds to the inner structure of time. The combination of spectra computed from temporal sequences with other techniques like granular synthesis [14] can be a powerful tool for experimentation in sound design. The temporal sequences and their spectra have been used as part of the structural basis of several works like Invertida Raíz for nine instrumentalists and computer , where the timbric integrations between the electronic and instrumental parts are produced with hybrid synthesis techniques, or Ad Matutinum for mixed choir where the harmonic fields are generated from the spectra of both vowels and temporal structures. The relationship at a perceptual level between the generated non periodic ordered rhythms and their associated spectra must be explored in a deeper level in order to expand our tools for algorithmic composition. The deterministic sequences are selfsimilar although,when they are included in a compositional context, this property is usually distorted at some level in the hierarchy . The analysis of deterministic and stochastic aperiodic ordered sequences can provide a unified basis from the point of view of the formalization process because it connects rhythmic, dynamic, harmonic and timbric striated spaces . Although without any type of a priori aesthetic consistency in the final results, the construction of the reference space discussed in this work opens new expression possibilities that can not be anticipated by intuitive explorations.

5 Acknowledgements

The NASA ADS Abstract Service and the SIMBAD database at CDS-Strasbourg, France, were used to access data about V346 Ori (D.F.M.Folha) and references.

6 References

1. Bombieri,E. and Taylor,J.E. 1986. Which Distributions of Matter Diffract ?. An initial investigation., Journal de Physique France, C3,Vol.47,19-28.
2. Cowell H. 1996. New Music Resources. Cambridge University Press.
3. Escudero,J.G.1993. La Tierra Ubérrima for string quartet. CD Agenda Edizioni Musicali 001.2004
4. Escudero,J.G.1995 Grammars for Icosahedral Danzer Tilings. Journal of Physics A: Mathematical and General. Vol.28, 5207-5215.
5. Escudero,J.G.1997. Estudio del Tiempo Iluminado II for computer. CD MasterVision 001. Concorso Internazionale di Composizione Elettronica "Pierre Schaeffer". SIAE.
6. Escudero,J.G.1999. Time and Frequency Structures within Selfsimilar Aperiodic Geometries. In H.G.Feichtinger and M.Doerfler (eds). Diderot Forum on Mathematics and Music. Oesterreichische Computer Gesellschaft Series. Viena,Vol.133, 145-152
7. Escudero,J.G.2000. Continuous and Discrete Fourier Spectra of Aperiodic Sequences for Sound Modeling. In D.Rocchesso and M.Signoretto (eds.) Proceedings Digital Audio Effects Workshop DAFX00 .Universitá degli Studi di Verona, .265-268
8. Escudero,J.G.2001 ET0L-Systems for Composite Dodecagonal Quasicrystal Patterns. International Journal of Modern Physics B. Vol.15 ,1165-1175.
9. Escudero,J.G.2003 A construction of inflation rules for Pisot octagonal tilings. International Journal of Modern Physics B. Vol.15 ,2925-2931.
10. Escudero,J.G. 2003. Fibonacci Sequences and the Multiperiodicity of the Variable Star UW Herculis.Chinese Journal of Astronomy and Astrophysics.Vol.3, 235-240
11.Godrèche, C., and Luck, J.M., 1992. Indexing the diffraction spectrum of a non-Pisot self-similar structure. Physical Review B.Vol.45, 176.
12. Hollander M and Solomyak B.2003. Two-symbol Pisot substitutions have pure discrete spectrum. Ergodic Theory and Dynamical Systems. Vol.23.533-540.
13. Pinheiro, F et al. 2003. Oscillations in the PMS Delta Scuti star V346 Ori. Astronomy and Astrophysics.Vol.399,271-274.
14. Roads, C. 2001. Microsound. The MIT Press. Cambridge.
Massachusetts.
15. Rozenberg G. and Salomaa A.1980. The Mathematical Theory of L Systems. Academic Press. New York.1980.
16. Stockhausen K,, 1963 ...wie die Zeit vergeht... Texte zur elektronischen und instrumentalen Musik. Bd.1. Dumont Buchverlag, Koeln.

A Self-Organizing Map Based Knowledge Discovery for Music Recommendation Systems

Shankar Vembu[1], Stephan Baumann[2]

[1] Technical University of Hamburg, Harburg
D-21071, Hamburg, Germany
shankar.vembu@tuhh.de
[2] German Research Center for Artificial Intelligence
Erwin Schrödinger Str., 67663 Kaiserslautern, Germany
stephan.baumann@dfki.de

Abstract. In this paper, we present an approach for musical artist recommendation based on Self-Organizing Maps (SOMs) of artist reviews from Amazon web site. The Amazon reviews for the artists are obtained using the Amazon web service interface and stored in the form of textual documents that form the basis for the formation of the SOMs. The idea is to spatially organize these textual documents wherein similar documents are located nearby. We make an attempt to exploit the similarities between different artist reviews to provide insights into similar artists that can be used in a recommendation service. We introduce the concept of a modified weighting scheme for text mining in the musical domain and demonstrate its role in improving the quality of the recommendations. Finally, we present results for a list of around 400 musical artists and validate them using recommendations from a popular recommendation service.

1 Introduction

Music similarity perception plays an important role in recommendation services for musical information. Given the huge amount of musical data available online, there is an increasing demand to provide quality-enriched services to allow access and browse musical content over the web. Various services are being offered by online music recommendation systems that use different approaches to compute music similarities [1], [2]. We present here in this paper an approach for musical artist recommendations using the album reviews for the artists available from the Amazon web site [3]. The basic idea of our approach is to spatially organize these reviews that are in the form of textual documents using an unsupervised learning algorithm called Self-Organizing Maps [4] and thus be able to give recommendations for similar artists by making use of the model built by the algorithm.

An approach to musical artist recommendations based on cultural metadata has been presented in [5], [6]. A comparison of audio-based music similarity measures in a peer-to-peer setting has been made in [7]. The use of SOMs in the field of text mining has been reported in [8], [9] and the WEBSOM project [10]. SOMs have also been used in the field of Music Information Retrieval for automatic analysis and organization of musical archives based on the audio content [11], [12] and in the SOM-enhanced JukeBox Music Digital Library project [13]. In this paper we investigate the use of SOMs for artist recommendations not based on the content-based audio analysis but by relying on textual information available in the web.

1.1 Amazon Web Service

The online shopping mall Amazon has made its products available through an interface [14] based on web service standards [15]. Using this interface, one is able to retrieve product information like artists/musicians, authors, ASIN and ISBN directly from the Amazon servers.

1.2 Self-Organizing Maps

The Self-Organizing Map (SOM) is an unsupervised learning algorithm used to visualize and interpret large high-dimensional data sets. The map consists of a regular grid of processing units called “neurons”. Each unit is associated with a model of some high dimensional observation represented by a feature vector. The map attempts to represent all the available observations with optimal accuracy using a restricted set of models. Map units that lie nearby on the grid are called *neighbors.* After the formation of a map for a particular data set, the model vectors are arranged in such a manner that nearby map units represents similar kind of data and distant map units represent different kinds of data. The reader is referred to [4] for a detailed description on this subject.

2 Self-Organizing Maps of Amazon Reviews

2.1 Textual Data Mining

The album reviews for artists can be accessed from the Amazon site using the Amazon Web Service interface [14] that is available as a standard development kit. Every artist can then be represented as a collection of his/her album reviews. The interface provides facilities to query a broad range of products from the Amazon site. The queries can be submitted either as a web service SOAP [15] message or using XML [16] over HTTP [16]. The latter uses URIs (Uniform Resource Indicators) with specific name/value pairs to invoke methods and processes within Amazon's Web Services framework. The returned response, which is a SOAP message in the former case and a well-formed XML document in the latter case, contains the complete product information.

The literature on Information Retrieval [17] provides techniques to preprocess and represent textual documents for mining operations. Preprocessing techniques include stripping unwanted characters/markup (e.g. HTML tags, punctuation, numbers, etc.) and removing common stop words (e.g. a, the, of, etc.). The documents are represented in the form of a bag-of-words where each document is considered as a point (or vector) in an n-dimensional Euclidean space where each dimension corresponds to a word (term) of the vocabulary. The i^{th} component d_i of the document vector expresses the number of times the word with index i occurs in the document, or a function of it. Furthermore, each word can be assigned a weight signifying its importance. Commonly used weighting strategy is the tf * idf (term frequency – inverted document frequency) scheme [18], where tf stands for term frequency within the document, and idf stands for the inverse of the number of documents in which the term appears. The scheme is based on the notion that a word that occurs frequently in the document but rarely in the rest of the collection is given more importance. A variant of the general tf * idf scheme that assigns a weight to every word is given by

$$w_{ij} = tf_{ij} * idf_i = tf_{ij} * \log_2 (N/ df_i) \tag{1}$$

where,
w_{ij} is the tf-idf weight for the i^{th} word in j^{th} document in a collection of N documents,
tf_{ij} is the term frequency of the i^{th} word in the j^{th} document and
$idf_i = \log_2 (N/df_i)$ is the inverse document frequency of the i^{th} word over the entire collection.

2.2 Modified Weighting Scheme for Text Mining in the Musical Domain

The tf * idf weighting scheme described above does not take into consideration any domain knowledge to determine the importance of a word. The words are given importance only based on the frequencies of their occurrence in the document as well as the collection. But when trying to find similarities between two documents in a musical context, it is desirable to exploit any domain knowledge that is inherently present in the documents. We propose one such mechanism to accomplish this by introducing the concept of a modified weighting scheme in the musical domain or context.

The album reviews for artists that are in plain English contain a few words that are more relevant and ought to be given more importance in the musical context. For example, consider an album review for a popular *rap* artist “Eminem” from the Amazon web site that says,

“…This is Eminem before he became the hottest rapper on Earth, when he was a lot more funny and creative. There are only like 3 songs that I don't like. The beats are just ok, but he makes up for it with his witty lyrics and tight flow. If you're into rap, you'll like it…”

Words like “rap” in the above example definitely say a lot about the artist being a rap artist and thus deserve more importance when the above document is represented. Eventually, one would like to find similarities between documents with words like the one described above being given more importance in comparison to other words occurring in the English language. Therefore, in addition to the weighting importance given to a word by the tf *idf scheme, it would be worthwhile to increase the weight of a word by a certain factor if it is pertaining to the musical domain. A few sample words that deserve more attention in a musical context are *samba, rap, hip-hop, metal, instrumental.* This would allow one to find similarities between documents not only by the occurrence frequencies of the words in them but also based on the musical importance of the words. Coming up with such a list of musical words, also plays an important role in the quality of the recommendations. We came up with such a word list of 324 from the genre taxonomies in [2]. The quality of document representation and thereby the results of the recommendations would definitely be increased if this modified importance weighting were incorporated into the system. The modified weighting scheme gives rise to a new weight for musical words that is given by

$$w^m_{ij} = tf^m_{ij} * idf^m_i = tf^m_{ij} * \log_2 (N/df^m_i) * \alpha \qquad (2)$$

where the superscript m indicates words belonging to the musical context and α is the weighting increment.

2.3 Map Formation

The preprocessed textual documents represented in the form of n-dimensional vectors can be used to train a Self-Organizing Map in an unsupervised way. The learning starts with a set of reference vectors also called the model vectors that are the actual map units of the network. As the learning proceeds, the model vectors gradually change or arrange themselves so as to approximate the input data space. The final arrangement is such that the model vectors that are nearby are similar to each other. The model vectors are usually constrained to a two-dimensional regular grid, and by virtue of the learning algorithm, follow the distribution of the data in a nonlinear fashion. The model vectors are fitted using a sequential regression process. Given a sample vector x(t) at iteration step t the model vector $m_i(t)$ with index *i* is adapted as follows:

$$m_i(t+1) = m_i(t) + h_{c(x),i}(t)[x(t) - m_i(t)], \qquad (3)$$

where the index of the "winner" model, c for the current sample is identified by the condition,

$$\forall i, \ \|x(t) - m_c(t)\| \leq \|x(t) - m_i(t)\|. \qquad (4)$$

$h_{c(x),i}(t)$ is called the *neighborhood function*, which acts as a smoothing kernel over the grid, centered at the "winner" model $m_c(t)$ of the current data sample. The neighborhood function is a decreasing function of the distance between the *i*th and *c*th nodes on the map grid. The regression is usually reiterated over all the available samples. Thus, with this unsupervised learning algorithm we can spatially arrange all the documents i.e. the album reviews of all the artists, resulting in a topological ordering of the artists. In addition to this, the SOM algorithm also obtains a clustering of the data onto the model vectors wherein the artists present in a particular cluster are similar to each other. In Fig. 1, if we consider map unit A, the artists in itself and its neighboring units show similarities.

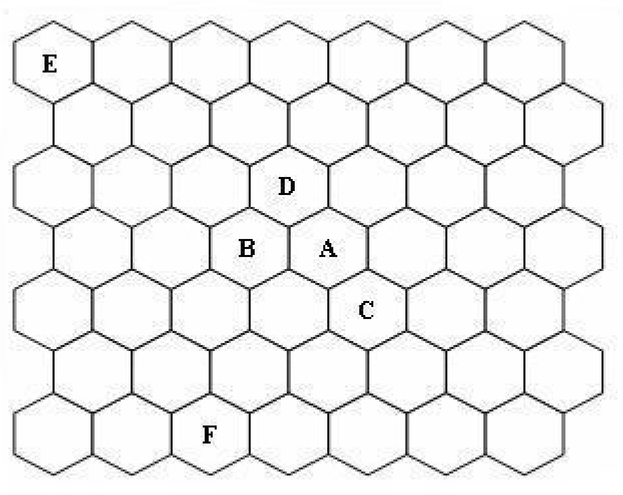


Fig. 1. A 7 * 7 SOM for the artist reviews with a hexagonal grid. Artists in map unit *A* are more similar to artists in nearby units like *B*, *C* or *D* than in far away units like *E* or *F*, which is by virtue of the self-organization of the map due to the learning algorithm

3 Experiments and Results

3.1 SOMs of the Whitman Artist List

We implemented the SOM based approach for artist recommendations described in this paper for the Whitman musical artist list [6] to build an SOM model for 398 popular artists. The most important album reviews for each artist of this list were obtained using the Amazon Web Service interface and stored as textual documents one for each artist. Each document was preprocessed and represented using standard techniques from the field of Information Retrieval [17]. Preprocessing techniques include stripping unwanted characters/markup (e.g. HTML tags, punctuation, numbers, etc.) and removing common stop words (e.g. a, the, of, etc.). The documents were then represented as a bag-of-words with each word being assigned a weight according to the tf-idf schema. We came up with a list of 324 musical words, to incorporate the concept of the modified weighting scheme described above in the musical context. If a document had words present in this list, its weight was increased by a factor of $\alpha = 4$. The weights were then normalized so that the size of the documents would not have any effect in their representation. A full-term indexing of the documents thus yielded a feature vector of 36,708 words for the data set of size 398. We then removed the words that were present in less than 5% and more than 90% of the collection as these words do not play an important role in classification [13]. This dramatically reduced the size of the feature vector to 3313. We thus had all the documents represented as a 398 * 3313 matrix.

For the map formation, we used the SOM toolbox [19] available for MATLAB.

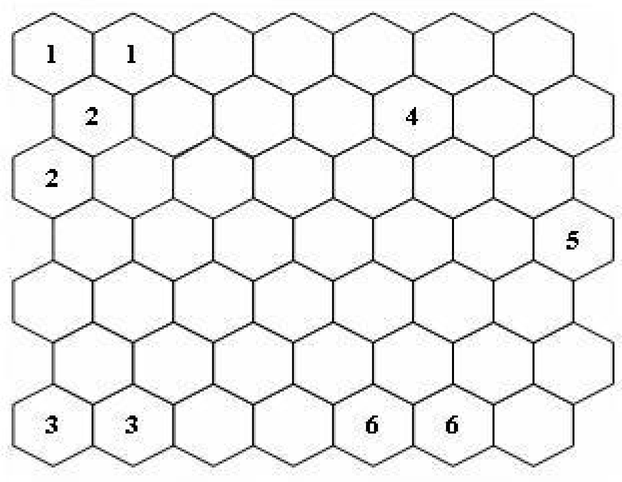


Fig. 2. Results for the Whitman artist list. The labels of the numbered map units and some of the artists present in them are given in Table 1

Table 1. Labels and artists for the numbered map units shown in Fig. 2

Map unit	Labels	Artists
1	Rap, Funk	Limp Bizkit, Korn, Linkin Park
2	Reggae, Rap	Bob Marley, Shaggy, Police, 311
3	Metal, Heavy	Metallica, Nirvana, Smashing Pumpkins
4	Techno, Remix	Alice Deejay, Prodigy, Moby, Chemical Brothers
5	Spanish, Latin	Ricky Martin, Enrique Iglesias
6	Rock, Roll, Blues	Dire Straits, Queen, Eric Clapton, Bob Dylan

We trained a Self-Organizing Map consisting of 49 map units in a hexagonal grid of size 7*7 for all the 398 artists using the 398 * 3313 matrix mentioned above. After the training phases, each map unit represented a model obtained from the collection of reviews in the high dimensional space. The entire data collection of 398 artists were divided according to the obtained map models on the two-dimensional grid and spatially organized based on their similarities.

3.2 SOM Labeling and Visualization

Labeling plays an important role in the visualization of the self-organizing map [20, 21]. We employed a simple labeling technique where a map unit is represented by a label or a keyword that has a higher weight, as calculated by the tf * idf weighting scheme, when compared to other words that appear in the map unit. The modified weighting scheme described in the previous sections also aided in labeling the map units. Since we increase the weight of a word that pertains to the musical context, many, if not all, of the labels that we obtained were from the musical list of words. This is indeed desirable when we are labeling an SOM of artists as we would like to see labels that are musical words like *rap, rock, metal, blues* and not plain English words. We present here the results of the SOM model described in the previous section with labeling. Fig. 2 shows a few distinct sections of the map with their respective labels and artists in Table 1. As can been seen from the results, we were able to obtain a clear categorization of artists based on different musical genres (rap, metal, rock, blues, techno etc.)

3.3 Validations

The results of our experiments were validated using Echocloud [1], a web-based music recommendation engine. Echocloud works by crawling peer-to-peer networks to capture users' file lists in order to discover correlations between musical artists. We would like to mention some limiting factors that made the validations difficult.

1. The Echocloud approach to finding similar artists differs widely from ours. Our approach entirely depends on the artist reviews from the Amazon site, whereas Echocloud is based on the notion that if two artists are found together in users' file lists, they are similar in some sense as having one tends to imply having the other [1].
2. The Echocloud builds the similarity model using a database of around 120k artists [1] in the so-called *open world* environment of users' music collection, whereas we performed experiments on the much-restricted Whitman artist list of around 400 artists.
3. Music similarity perception is *subjective* in nature. Even though we do not present a *subjective* validation in this paper, we believe that such a validation is more efficient for experiments related to music similarity perception. We presented such an approach in an ecological environment in [22].

We compare the Top 10 recommendations from Echocloud with our Top 10 recommendations for all the artists. We also compare Echocloud recommendations with the

Table 2. Validations of our results for 398 artists with Echocloud's Top 10 recommendations without modified weighting scheme

Comparisons	Total number of recommendation matches	Average recommendation match in percentage
Top 10	482 / 3980	12.1 %
3 BMUs	685 / 3980	17.2 %
5 BMUs	982 / 3980	24.7 %

Table 3. Validations of our results for 398 artists with Echoclould's Top 10 recommendations with modified weighting scheme

Comparisons	Total number of recommendation matches	Average recommendation match in percentage
Top 10	493 / 3980	12.4 %
3 BMUs	785 / 3980	19.7 %
5 BMUs	1038 / 3980	26.1 %

artists that are present in the 3 and 5 Best Matching Units (BMUs) of the artist in question. A Best Matching Unit for an artist i.e. textual document for the review is the SOM map unit that best models it. The Euclidean distance measure is used in finding the BMUs for an artist.

As can be seen from the results in Tables 2 and 3, the quality of the recommendations increases as we move up from the Top 10 comparisons to 5 BMUs. This leads to an interesting observation that even though all the Echocloud recommendations do not match with our Top 10 recommendations, some of them do certainly lie in the nearby regions namely the 3 or 5 BMUs, which is by virtue of the map's self organization. Most importantly, we were able to see an increase in the quality of recommendations when the modified weighting scheme that exploits the musical domain knowledge was incorporated into the system. We believe that the list of 324 words pertaining to the musical context that was used for this can play an important role in a recommendation service to yield better results, even though there is human intervention in coming up with such a list.

4 MYMO: A Prototypical Application for Mobile Music Information Retrieval

4.1 Existing Framework

We came up with a mobile music recommendation engine called MYMO that uses internal as well as external services to provide recommendations for similar songs and artists. The song similarities are computed internally using a trimodal global similarity measure to provide the user a set of similar songs given an anchor song. This measure is realized as a weighted linear combination of three different local similarity metrics, namely sounds-alike similarity, similarity of lyrics and cultural or stylistic similarity.

$$S = wso^* Sso + wly^* Sly + wst^* Sst \tag{5}$$

where
Sso: sounds-alike similarity
Sly: similarity of lyrics
Sst: similarity by style/cultural aspects and
wso, wly, wst are the respective weights.

For a detailed description of the local similarity metrics, the reader is referred to [22]. For the artist similarities, we interfaced with an external recommendation service, Echocloud [1]. The engine was realized as a web based solution (Fig. 3). The web site can be displayed on a small screen typical of PDA (personal digital assistant) or PIM (personal information management) devices. To allow the user subjective and interactive feedback, a virtual joystick was included that can be easily accessed using the pen of the PDA. The recommendation engine uses the song similarity measure described above. The position of the joystick has a direct influence on the individual weights in the linear combination. In this way the individual users can select different settings and find their favorite combination.

4.2 Integrating SOM Based Artist Recommendation Approach into MYMO

The SOM based approach for musical artist recommendations described in the previous sections can be integrated into our existing framework of mobile music recommendation service to provide recommendations for similar artists. This can be used as a replacement for the external Echocloud service that is currently being used to provide artist recommendations. The SOMs provide a spatial representation of the high dimensional artist reviews. A model can thus be formed for a set of artists where similar artists are located nearby spatially and dissimilar artists lie farther away. Furthermore, it is not mandatory that the queried artist be present in the model formed by the SOMs. The review for such a query can be obtained online from the Amazon site and the best-matching unit (BMU) for this review can be found from the model. The artists present in this BMU are therefore similar to the queried artist.

5 Conclusions and Discussions

We have presented in this paper a novel approach to musical artist recommendations using Amazon reviews and Self-Organizing Maps. We introduced the concept of a modified weighting scheme for text mining in the musical domain and demonstrated its role in improving the quality of recommendations. We presented results for the Whitman artist list of around 400 musical artists and validated them with the recommendations from Echocloud, a web based recommendation service for musical artists.

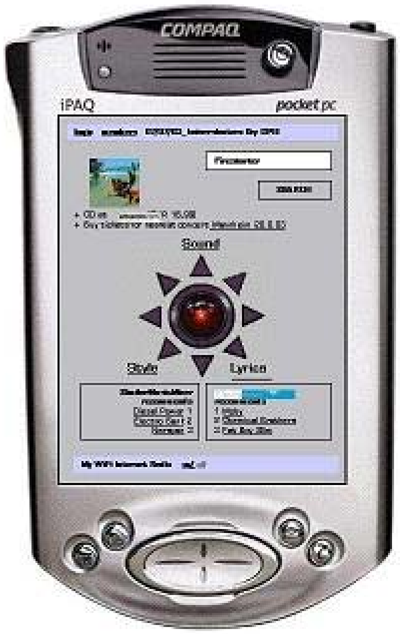


Fig. 3. MYMO: Mobile Music Recommendation Engine

We also demonstrated how this approach can be used in a music recommendation service. Exploiting the semantics of the language behind the artist reviews by using Natural Language Processing techniques could be a possible future work. The goal is to extract as much relevant information as possible from the artist reviews by using the state-of-the-art technologies from the fields of Information Retrieval and Natural Language Processing. We believe that the approach presented in this paper would be well suited for use in a recommendation service to provide quality services to all the music lovers of the world.

6 Acknowledgements

We wish to thank the *Stiftung Rheinland-Pfalz für Innovation* for supporting this work.

References

1. http://www.echocloud.net
2. http://www.allmusic.com
3. http://www.amazon.com

4. Kohonen, T.: *Self-Organizing Maps*, Springer, Berlin, Heidelberg, 1995
5. Baumann, S., Hummel, O.: “Using Cultural Metadata for Artist Recommendation”, Proc. Of the Third Conference on Web Delivering of Music (WEDELMUSIC 2003), Leeds, UK, September, 2003
6. Whitman, B., Lawrence, S.: “Inferring Descriptions and Similarity for Music from Community Metadata”, Proceedings of the 2002 ICMC, Göteborg, Sweden, 16-21 Sep.2002, pp 591-598
7. Baumann, S., Pohle, T.: “A Comparison of Music Similarity Measures for a P2P Application”, Proc. of the 6th Int. Conference on Digital Audio Effects (DAFX-03), London, UK, September 8-11, 2003
8. Lagus, K.: “Text retrieval using self-organized document maps”, Technical Report A61, Helsinki University of Technology, Laboratory of Computer and Information Science, ISBN 951-22-5145-0, 2000
9. Merkl, D. and Rauber, A.: “Document Classification with Unsupervised Neural Networks”, in Fabio Crestani, Gabriella Pasi (Eds.), Soft Computing in Information Retrieval, Physica Verlag & Co, ISBN:3790812994, Germany, 2000, pp. 102 - 121
10. Lagus, K.,: “Text Mining with the WEBSOM” Acta Polytechnica Scandinavica, Mathematics and Computing Series no. 110, Espoo 2000, 54 pp. D.Sc.(Tech) Thesis, Helsinki University of Technology, Finland, 2000
11. Rauber, A. and Frühwirth, M.: “Automatically Analyzing and Organizing Music Archives”, In Proceedings of the 5th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2001), Darmstadt, Germany, Sept. 4-8 2001, Springer Lecture Notes in Computer Science, Springer, 2001
12. Pampalk, E., Dixon, S. and Widmer, G.: “Exploring Music Collections by Browsing Different Views”, Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR'03), Baltimore, MD, October 26-30, 2003, pp 201-208
13. Rauber, A., Pampalk, E. and Merkl, D.: “The SOM-enhanced JukeBox: Organization and Visualization of Music Collections based on Perceptual Models”, In: Journal of New Music Research (JNMR), 32(2):193-210, Swets and Zeitlinger, June 2003
14. Amazon Web Service, http://www.amazon.com/gp/aws/landing.html
15. Web Services Activity, http://www.w3.org/2002/ws/
16. World Wide Web Consortium, http://www.w3.org/
17. Ricardo Baeza-Yates and Berthier Ribeiro-Neto: *Modern Information Retrieval*, Addison-Wesley, 1999
18. Salton, G. and Buckley, C.: “Term weighting approaches in automatic text retrieval”, Tech. Rep. 87-881, Cornell University, Department of Computer Science, Ithaca, NY, 1987
19. SOM Toolbox for MATLAB, Helsinki University of Technology, Laboratory of Computer and Information Science, Neural Networks Research Centre
20. Rauber, A.: “LabelSOM: On the Labeling of Self-Organizing Maps”, Proceedings of the International Joint Conference on Neural Networks (IJCNN'99), Washington, DC, July 10 - 16, 1999
21. Lagus, K. and Kaski, S.: “Keyword selection method for characterizing text document maps”, Proceedings of ICANN99, Ninth International Conference on Artificial Neural Networks volume 1, pages 371-376, IEE, London, 1999
22. Baumann, S. and John Halloran: “An ecological approach to multimodal subjective music similarity perception”, to appear in Proc. Of the First International Conf. On Interdisciplinary Musicology, Graz 15.-18.April, 2004

Internet Archive of Electronic Music IAEM internet Audio Rendering System iARS

Christopher Frauenberger and Winfried Ritsch

Institute of Electronic Music and Acoustics,
University of Music and Dramatic Arts Graz
Inffeldgasse 10/3, 8010 Graz, Austria
{frauenberger,ritsch}@iem.at
http://iem.at

Abstract. The Internet Archive for Electronic Music (IAEM) is intended to be a platform to access an extensive and distributed archive of electronic music. It combines collaborative tools, real time signal processing on the client side and the content of the archive with the concept of learning sequences to form a powerful teaching, research and publishing tool. The internet Audio Rendering System (iARS) is a client browser extension which is part of the IAEM system. It extends a web-browser with a flexible real time audio processing and OpenGL graphic display capability supporting multi-channel processing and streaming. This enables users of the system to perceive multi-track recordings showing their intentional acoustical context in virtual concert halls. These environments may be used embedded in learning sequences for teaching, opening new opportunities in music education.

1 Introduction

The "Internet Archive of Electronic Music" (IAEM) provides access to a distributed archive of electronic music for the purpose of education and research. It combines state-of-the-art Internet tools for information exchange and collaboration with multimedia real-time processing and virtual concert simulations for the exploration of works of electronic music and their interpretation.

With the integration of eduPlone the IAEM system provides an easy to use platform for lecturers in which reusable learning elements can be incorporated into learning sequences for online courses. All elements are standard compliant according to the IMS Global Learning Consortium which is considered to be the most wide spread standard for eLearning content.

The unique character of electronic music as eLearning content is considered by the development of the internet Audio Rendering System (iARS). iARS is a browser plugin based on the well established computer music program Pure Data (by Miller Puckette) allowing users to interactively apply sound processing on online music streams. This opens new possibilities in music education like online courses for spatial audio mixing or sound restoration. iARS in combination with an extensive library of electronic music and the possibilities of eduPlone learning sequences will have great impact on the way courses can be organised at music universities.

Beyond the scope of education the IAEM system is intended to serve the community of electronic music artists as a platform for publishing their work and get qualified feedback. The realisation of the system includes tasks within very different fields: legal aspects, technical challenges, the didactic approach and the organisational framework.

This paper addresses these aspects. The subsequent sections discuss the architecture of the IAEM system and its main components: The content database as storage for the music content, the IAEM portal integrating the content storages, eLearning methodologies, collaboration tools and the iARS browser plugin. Furthermore this section will address security issues. Subsequently, iARS is presented in detail showing the approach and its realisation. Finally, we draw a conclusion, summarise the paper and present future plans about IAEM.

2 Architecture

The architecture of the IAEM system is a classical server-client approach with distributed databases as back-end data source. But there is a significant difference: clients may also connect to the content databases directly. Figure 1 illustrates the approach.

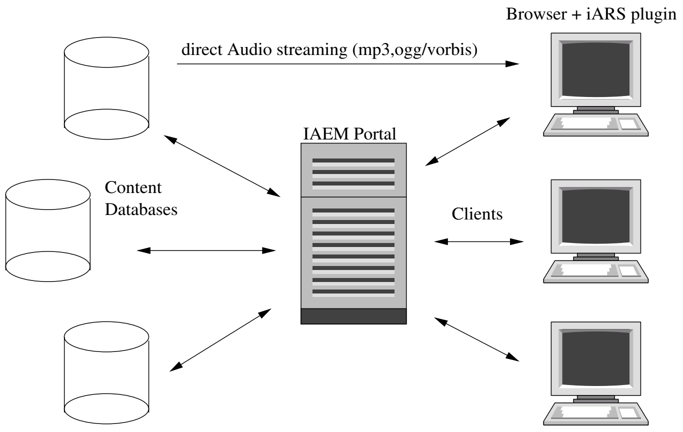


Fig. 1. Basic structure of the IAEM including client terminals

At the core is the IAEM portal server which provides a content management system. This portal may connect to a list of content databases where the compositions are stored along with additional meta-data as usual in common music libraries (composer, artists etc). The portal can process search-queries on the data in order to present results to the user via the web interface. The user can browse through the information, attend to discussions or learning sequences and select a work of music along with an audio rendering algorithm for listening. Upon the user's requests to receive a piece of music, the iARS browser plugin starts at his or her personal browser. It loads the chosen algorithm and connects to the content database to receive the requested music as an audio stream. The plugin also provides graphical user interface in the browser window through which the behaviour of the audio rendering algorithm can be altered online (i.e. during operation). This offers users an interactive experience in music retrieval via the Internet.

The direct connection between the client and the content database decreases the performance requirements for the IAEM portal, since clients retrieve data without occupying bandwidth or other resources from the portal.

3 The Content Databases

An IAEM content database system consists of four main components. The database itself stores references to the audio data in the file-system and the additional meta-data. This database can be queried by the IAEM portal through a standard SQL interface. A control software at the content database server is is responsible for managing the streaming server. It executes commands received from the portal via a XML-RPC interface [1]. With these commands the portal can initialise a stream, start or pause it and remove the streaming mountpoint. It also controls the security layer. It will strengthen the security and peer authenticity by certificates and SSL tunnel transmission.

The migration of an initial set of audio data into the system falls under the responsibility of the operator and/or the partner institution. Subsequently all users of the portal may contribute to its content, with an integrated reviewing process ensuring the quality of information. The IAEM system provides a good reason for institutions to digitise their music archives, especially multi-track works. Multi-channel and audio rendering capabilities of the system create a realistic reproduction of such works.

3.1 Streaming

In order to provide multi-channel capabilities the IAEM content database system needs to employ a streaming server technology which supports multi-channel audio formats. Ogg vorbis is a new and free compressing audio data format for encoding mid to high quality audio at variable bitrates from 16 to 128 kbps/channel. Since version 1.0 rc1 this standard also provides channel coupling mechanisms designed to reduce effective bitrate by both eliminating interchannel redundancy and eliminating stereo image information labelled inaudible or undesirable according to spatial psychoacoustic models.

The streaming server of choice is IceCast 2, since it supports both the newer ogg vorbis format and mp3. For setting up a stream between a content database server and the client iARS plugin, the IAEM portal generates a random identifier for the stream and initialises a streaming mountpoint at the content database server. It also provides the information to the iARS browser plugin which is then able to connect directly and receive the stream. These identifiers are unique and only available at the time an authenticated user requests a stream; this prevents other clients from connecting to a stream which was not created for them.

The future security concept is intended to prevent interceptions of the stream and guarantee peer authenticity of the streaming partners. Based on a IAEM certificate authority a certificate is issued for every user and content database. These certificates are verified at the beginning of each streaming event and will authenticate the streaming partners. Furthermore, it is planned to use a SSL tunnel for the streaming connection so that sensitive data may not be intercepted.

3.2 Database

Along with references to the audio data the content database contains meta-data related to the compositions. The design is based on a relational database structure and is similar to commonly used library systems, but simplified to suit the given requirements. The interface for portal queries is a standard SQL command set. Meta-data include references to a composer database, texts or lyrics, scores and other musicological observations.

Each record has a unique reference within all databases in the IAEM-Network. Content can therefore be referenced unambiguously even from multiple portals. Furthermore portal content, such as documents, discussions, mails can be associated to database content by referencing this unique ID.

The following figure illustrated the database layout:

Figure 2 shows database tables and their relations. Besides the customary *Person* table holding composers and artists, the *Piece* table is the core of the structure. It represents the abstract idea of a work of music and is linked to a certain recording through a *Performance*. Digitised music data very often refers to an analog source. This is realised by *RecordingMaster* - the origin which resulted from a performance - and *RecordingCopy* which is the actual digital version available in the database. The *Material* table is a generic container for all material associated with a *Piece* or any *Person*. This might be a score or biography and may be of any type covered by *MaterialType*. All materials and digital copies of any recording is restricted in use by a license stored in the *License* table. These license texts cover all possible restrictions and are the basis for the portal to decide whether to grant user permissions for the source.

4 The IAEM Portal

The IAEM portal is a content management system with various collaboration tools with features to drive the iARS plugin and to query the content database systems. The chosen framework is Zope [2] extended with CMS[1] and Plone [3].

[1] Content Management System

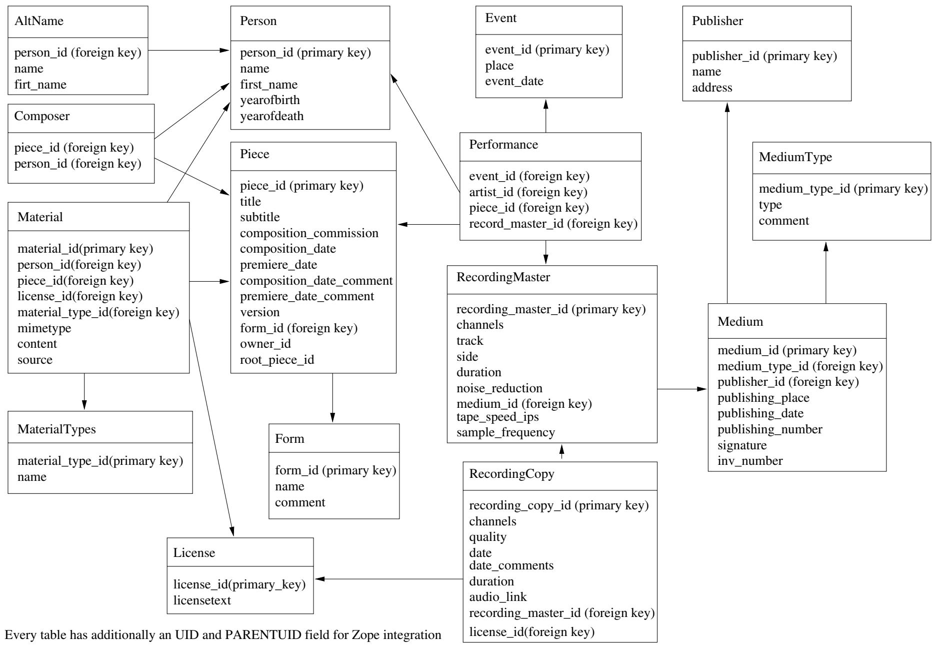


Fig. 2. The database layout for IAEM content databases

Object orientated representation of content makes the system expandable for information retrieval plugins extending the portal's functionality. For integrating learning sequences into the portal eduPlone was chosen as the state-of-the-art eLearning platform.

The data presented by the portal is legally sensitive so that a secure authentication method is compulsory. The Zope system provides an LDAP[2] authentication product with which the user must log in before the portal can be used. This allows also a personalised environment with user defined folders and content. The rights can be set for every single user so that the access to music and meta-data can be clearly determined to prevent any legal conflicts.

Collaboration tools are integrated to facilitate the communications between the users. It is hoped that vital discussions and information exchange will enhance the content of the portal. Mailing-lists, discussion forums, information agents and other common collaboration tools are available to make such exchanges possible.

For publishing, the portal features uploading to a content database. The access rights for user published data can be set by the author via the portal. For searching the content databases a single line search exists, as well as, a more complex advanced searching and cataloguing facilities.

[2] Lightweight Directory Access Protocol

The didactic value for teaching is given by the integration of the tools into eLearning courses. eduPlone is a product especially designed for the use with the Plone CMS system and provides learning sequences for designing eLearning courses [4]. For example: lecturers may design a course for digital recording techniques by making raw material (a piece of not post-produced music) available via the portal. In various learning sequences the students can then be asked to solve problems such as finding a specific loudspeaker arrangement for the recording. Students could play and explore within iARS and save their settings to submit their results. Along with such sequences a vital exchange between the students and the lecturer can take place by employing the collaboration tools integrated into the system.

5 The iARS Browser Extension

iARS (internet Audio Rendering System) is a browser plugin extending the browser's capabilities with a flexible audio rendering machine. It can be invoked by an "object" tag within web pages. The signal processing is done by the Pure Data programme which is launched by the plugin and remote controlled via a XML-RPC interface. The processing algorithm can be defined as a regular Pd patch along with a graphical representation of the patch. This is done using a IDL (Interface Description Language). Following such description the plugin draws controls into the browser window with which the behaviour of the algorithm can be altered.

iARS implements the Netscape Gecko Plugin API to communicate with the browser. During the initialisation process the plugin checks for running instances of Pd and launches an instance if needed. The plugin control block remotely controls the Pd programme and builds the graphical representation of the loaded patch. The Pd program is launched with externals which extend the capabilities of Pd for XML-RPC communication and audio streaming. The GEM library is used to draw real time computer graphics to an assigned window area using openGL.

5.1 Operation

The plugin is launched by using the "object" tag embedded in regular HTML code. A MIME type is registered by the plugin at the browser which refers to the the data type associated. The following listing shows an example HTML code for embedding iARS objects.

Listing 1.1. Embedded object tag

```
<html>
 ...
<body>
<OBJECT type="application/iARS"/>
```

```
 <param name="patch" value="http://iaem.at/amb.pd">
 <param name="gui" value="http://iaem.at/ambgui.xml">
 <param name="stream" value="http://db1.at:8888/NHS271/">
 <param name="extras" value="http://iaem.at/extras.zip">
</OBJECT>
...
</body></html>
```

The object's application/iARS MIME type causes the browser to launch iARS. A window handle provided by the browser is assigned to the plugin for its graphical representation. The object tag must provide the following information as parameters for iARS: The *stream* field determines the URI of the requested audio stream, "patch" and "gui" both in URI format assign the Pd patch to be loaded and its graphical representation. A zip archive of extras may be specified to provide the patch with additional resources like abstractions or Pd externals.

The Gem external of Pd allows the plugin to draw virtual concert situations into the browser window. It can be used, for example, to show the position of virtual surround loudspeakers and even allow altering their position in the virtual room.

5.2 Pure Data

Pure Data is a real time signal processing tool for customary PCs [5]. There are many extension libraries available for Pd. The two main extensions developed for the IAEM project are the XML-RPC interface and the streaming external. The main advantage of using Pd as the processing core application is the multitude of existing patches. Due to generic approach of the plugin these patches can be reused with only minor adjustments.

The XML-RPC interface to the Pd programme is intended to become a comfortable standard of remote controlling the application. It is possible to load and close patches, but also to communicate with every single element of a patch. There are mechanisms to bind callback functions to symbols so that an event triggered communication - desirable for GUIs - is possible.

5.3 Graphical Representation

The graphical representation of a patch is not defined within the patch. This allows the reuse of existing patches and the definition of a interface description language (IDL) more suitable for our application than the existing. The implementation of the controls was made using Trolltech's Qt toolkit [6].

Within the IDL file several controls are defined which are bound to elements of the Pd patch. If either the user interacts by changing the value in the GUI or the patch alters the value, the counterpart is informed. In this way, parameters of the patch can be altered and values can be displayed correctly. The following listing shows an example of a IDL file describing a graphical representation of a Pd patch.

Listing 1.2. Interface Description Language Example

```
<?xml version="1.0"?>
<!DOCTYPE interface SYSTEM "idl.dtd">
<interface>
<author>Christopher Frauenberger</author>
<patch>Ambisonic 3D</patch>
<version>1.0</version>
<input name="stream"/>
<group name="Example" orientation="horizontal">
  <vslider name="Volume" bind="volume"
           min="0" max="100" value="10"/>
  <levelmeter name="Left" bind="vul"
              min="-100" max="0" value="0"/>
  <levelmeter name="Right" bind="vur"
              min="-100" max="0" value="0"/>
  <group name="GEM" orientation="vertical">
    <onoff name="GEM Window" bind="gemwindow" value="1"/>
    <onoff name="OpenGL" bind="draw" value="1"/>
    <hslider name="Rotation"
             bind="rotate" min="0" max="180"/>
  </group>
</group>
</interface>
```

In the above example a simple interface is built with a vertical slider for the volume, two levelmeters, another slider along with two buttons. The group tag allows the user interface elements to be grouped together in a frame. The description shown above results in a graphical user interface shown in figure 3.

All possible tags and their relations are described in the document type definition “idl.dtd”.

6 Conclusion

The proposed system combines very recent technologies to a powerful research and lecturing tool. All components were designed to be flexible and generic. The distributed architecture allows different partners to collaborate in providing their clients with a comprehensive library of electronic music.

The iARS plugin is an approach to introduce real-time audio rendering to the world of web applications. The underlying Pd programme was chosen because of its performance and availability for a wide range of platforms.

Future work will definitely need to proof the concept by usability tests. The portal and its components will be redesigned on the basis of the results of such studies.

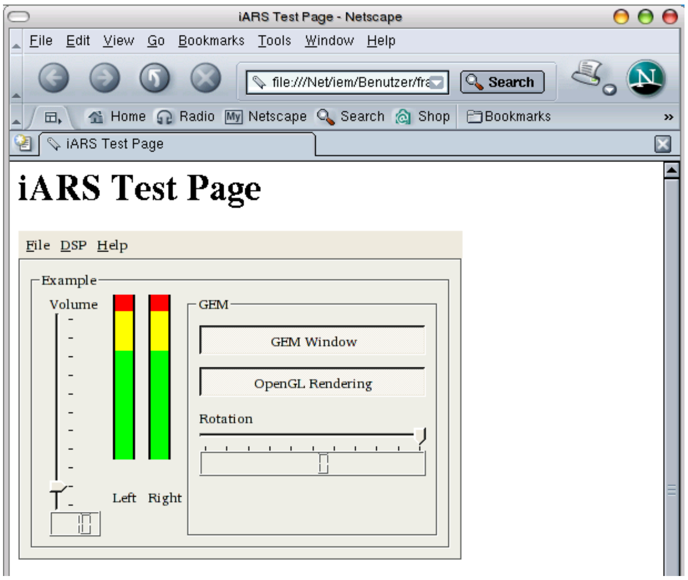

iARS Test Page - Netscape
File Edit View Go Bookmarks Tools Window Help
file:///Net/iem/Benutzer/fra
Search
Home Radio Netscape Search Shop Bookmarks
iARS Test Page
iARS Test Page
File DSP Help
Example
Volume
Left Right
GEM
GEM Window
OpenGL Rendering
Rotation

Fig. 3. Screenshot of the interface described by the IDL example above

7 Acknowledgement

This project was kindly funded by the Austrian Federal Ministry for Education, Science and Culture within the "New Media in teaching at universities and polytechnics in Austria" project framework.

References

[1] D. Winer, "Xml-rpc specification," Tech. Rep., Userland, xml-rpc.com, 1999, http://www.xmlrpc.com.
[2] Zope, *The Zope Book*, 2004, http://zope.org/Documentation/Books/ZopeBook/2_6Edition/.
[3] Plone, *The Plone Book*, 2004, http://plone.org/documentation/book.
[4] "eduplone concepts," http://www.eduplone.net/concepts/, 2004.
[5] Miller Puckette, *Pd Documentation*, 2003, http://crca.ucsd.edu/~msp/.
[6] Trolltech Inc., *Qt Reference Manual*, 2003, http://doc.trolltech.com/3.1/.

Handel, a *Free-Hands* Gesture Recognition System

Leonello Tarabella

computerART project of ISTI / CNR - Pisa, Italy
Research Area of the Italian National Council of Research
leonello.tarabella@isti.cnr.it
http://tarabella.isti.cnr.it

Abstract. I describe here a real-time vision-based gesture recognition system used in interactive computer music performances. The performer moves his hands in a video-camera capture area, the camera sends the signal to a video digitizer card plugged in a laptop computer. By processing the reconstructed images of the performer's hands in movement the computer detects x-y positions, shape (posture) and angle of rotation of both the hands. Data extracted from image analysis every frame is used for controlling real-time interactive computer music performances. Two approaches, one more formal the other really operative, are presented.

1 Introduction

Modern human-computer interfaces are extremely rich, incorporating devices such as keyboards and mouse and a wealth of advanced media types: sound, video, animated graphics etc. In addition, advanced interaction strategies are being considered. A good example of that is gesture interaction [1,2] where actions of a system are controlled by a series of hand positions or postures. The term multi-modal is often associated with such interfaces to emphasize that the combined use of multiple modes of perception (e.g. visual and tactile) is relevant to the user's interface [3].

An interface, in a conventional sense, might comprise a window system and a mouse through which interaction is possible by taking into consideration specific areas of windows in the computer display. The system allows for very few degrees of freedom at the same time because usually maps the two-dimensional cursor location to *do-it* commands.

The problem with this approach is that it analyses human performance in terms of encoded rules thus forcing a specific behavior on the performer. With this setting, the user is likely to fill a sense of technological awareness and he tends to perceive the machine as his primary interacting partner. In this way each act is performed to communicate the intended information with great details implicitly inhibiting the potential of human effectors in enriching the semantic content of the information to be communicated [4].

Artistic performers usually needs many degrees of freedom to control at the same time in order to communicate their emotions for giving expression to music based on technology. This can now be achieved by including a computer in the loop between the human physical actions and the musical response, while addressing two basic principles: - holding nothing at all: controllers respond to body position and motion without requiring anything to be grasped or to be worn connected with wires; - sensitive space: controllers sense the player to give the person the strong feeling of being *bathed* in sound [5].

Moving along the guidelines highlighted so far, I have started developing human-computer interfaces by using non-intrusive devices and systems based on the remote sensing of postures of the hands and more generally of the human body [6].

1.1 Background

At the beginning of the '90s I started to tackle the fascinating realm of the real-time control of digital sound and, together with other researchers and collaborators of C.N.R. in Pisa, I realized a number of devices and systems based on the infrared (IR) and the real-time analysis of video captured images technologies: TwinTowers, Light Baton, UV-Stick, Imaginary Piano and PAGe system. The *TwinTowers* is an electronic device based on IR technology consisting of 2 groups of four elements arranged as the vertical edge of two parallelepipeds. After having presented this device many times at technological and artistic level [7,8], I recently developed a new version, also named PalmDriver, consisting of up 8 groups of 4 elements, which works as a standalone device properly equipped with a MIDI OUT port.

Image processing technology has been used for realizing the other systems. The same hardware and the same strategy have been used for implementing them all. A CCD camera is connected to a video grabber card and, whatever the system, the digital image to be analyzed consists of the reconstructed image by means of an algorithm which filters (that is, accepts) those pixels whose luminance is greater than a predefined threshold. Although this algorithm would be not applicable to a generality of images, it is precise enough to distinguish the luminance values of those pixels corresponding to the hands from the rest of the scene. Besides, in order to improve the robustness of the method, the performer dresses in black and has at his shoulders a black background.

The *Light Baton* system has an on-board light LED source on the conductor's baton tip powered by a battery placed in the cork handle. Implemented for conducting a computerized orchestra, this system recognizes those movements of the baton made by the conductor during a live performance that conform to international standards [9].

The *UV-Stick* (and the systems reported in the following) works on the basis of images of object or the hands themselves captured by the CCD camera and lit by a source light placed where usually the camera is placed. In particular, in the UV-stick the source light is an Ultra Violet lamp that gives the stick (a Plexiglas tube 50cm long and 3cm diameter) a suggestive visual impact of a *laser sword*. The extreme points of the stick are used for detecting the barycenter x-y position and its angular rotation [10].

In the *Imaginary Piano* a pianist sits as usual on a piano stool and takes into account an imaginary line at the height where the keyboard of a real piano usually lies:

when a finger, or a hand, crosses that line downward, the systems reports proper information regarding the *key number* and a specific message is issued accordingly to *where* and *how fast* the line has been crossed [11].

In the *PAGe* (Painting by Aerial Gesture) system, positions and movements of a performer's hands are recognized in an wide vertical plane; the performer acts as a painter who uses his hands for selecting colors and nuances of color and for actually drawing a picture; movements are performed in the air and the resulting picture is projected on a large video-screen. Beside, special gestures trigger preset sounds which makes operative the paradigm of *synaesthesy in art* [12,13].

These gesture recognition systems produce data-streams used for controlling sound and graphics in real-time. To map information to sound, that is to define how to link data coming from gesture recognition systems to algorithms that generate music, it's up to the composer himself and is related to a specific composition [14].

After the experience gained from the realization of the above-described systems, I tried a formal approach for realizing a general-purpose system able to recognize shape, position and movement of the hands. The basic idea of this approach to gesture recognition was derived from a paper by V.Cappellini [15] based on the Fourier Transform and developed for recognizing bolts and tools sliding on a conveyor belt to be selected and picked up by a mechanical arm.

Anyway, since this method (presented at the ICMC97 [16] and here briefly reported) although stable and elegant, results rather *time consuming* and therefore not fully suited for real-time application such as interactive controlled computer multimedia performances. So, I developed a less formal but more practical and really operative system for the purpose. I'll describe it in paragraph n.3.

2 Formal Approach

The digitized image coming from the camera is transformed into a binary matrix where 1's represent those points $p(x_i,y_j)$ whose luminance level is greater than a predefined threshold. The *barycenter*, i.e. the center of mass (x_c,y_c), is given by the weighted mean of rows and columns on the binary matrix as follows

$$x_c = \frac{\sum_i \left(i \times \sum_j c_{i,j}\right)}{\sum_{i,j} c_{i,j}} \qquad y_c = \frac{\sum_j \left(j \times \sum_i c_{i,j}\right)}{\sum_{i,j} c_{i,j}}$$

where x_c and y_c are the coordinates of the center of mass and $c_{i,j}$ is the (*i*-th, *j*-th) component of the binary matrix so that: $\sum_j c_{i,j}$ and $\sum_i c_{i,j}$ are the count of pixels valued 1 in the *i*-th row and in the *j*-th column respectively and $\sum_{i,j} c_{i,j}$ is the total number of pixels valued 1 representing the hand.

Next step consists in constructing a *one-period-signal* by the distances from the barycenter of those points along the contour taken on radii at predefined angular steps. For searching the second point of each segment (first one being always the barycenter) program searches on lines $Y=mX+q$ for the most distant point of value=1, that is *white*; *m* is the angular coefficient of radius which changes with step $\Delta\delta$ corresponding to the virtual *sampling rate frequency*.

Since in general the posture of the hand generates non-convex figures, the scanning algorithm just described produces signals corresponding to a *palmed* hand (like ducks feet). However, as experimentally verified, this approximation does not affect the analysis results. More critical is the choice of step $\Delta\delta$: when too large the algorithm produces aliased signals and when too small great amount of computation is requested for the FFT.

Experimentally good values have been found to be $\Delta\vartheta = \frac{2\pi}{32}$ and $\Delta\vartheta = \frac{2\pi}{64}$.

The following figure show two different typical postures of the hands, their corresponding one-period-signals constructed as described and the resulting harmonic spectrum computed by the FFT algorithm.

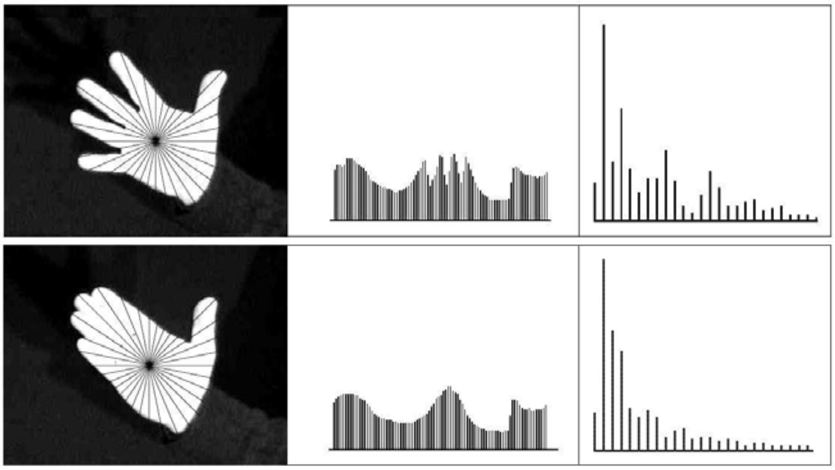


Fig. 1. Postures, one-period-signals and FFT analysis

The harmonic spectrum characterizes very well the posture of the hands and, furthermore, has the very important property of invariance with respect to both rotation and dimension (which changes with the distance from the camera). The result of FFT is input to the actual recognizer that employs an algorithm measuring the distance between n-dimensional vectors.

Let the vector h=(h_1,....h_n) represent the harmonic spectrum derived from the feature associated to a hand's posture and let C be the set of vectors, each representing the harmonic spectrum of a corresponding posture, previously recorded while training the system. The recognizer selects a vector c* from the set C as the harmonic spectrum representative of h such that for all $c=(c_1,....c_n) \in C$ it holds that $\|h - c^*\|_2 \leq \|h - c\|_2$ with $\| \ \|_2$ denoting the L2-norm. Rotation comes from the phase spectrum: in this case only the first component is meaningful since the higher components are simply multiples of the first one.

3 Operative Approach

As I said I developed a new system that gets information from postures and positions of the hands. This is less formal but, on the other *hand*, more flexible, faster and more usable thanks to an high number of parameters put at disposal at the same time. Once again, hardware and methodology are based on a video camera which captures the images of the performer's hands, the performer dressing in black on a black background.

Now, instead of recognizing the shape of the hands, it's a matter of taking into consideration the rectangle that *frames* the white spots of the hands. In this way parametric values provided by the real time analysis on the reconstructed dot images deal with dimensions and x,y-coordinates of the center.

The video capture area is converted into a matrix of pixels that is then scanned and analyzed. In the same manner as it happens in the well-known BigEye [17] application, it's possible to define sub-zones where to apply the analysis process. This has two main advantages: the process is faster and at the same time it solves the problem of the presence of the performer's face. Without this facility, a complex and not fully reliable algorithm for filtering out the performer's face should be implemented. The sub-zones where to run the analysis can be dynamically defined.

The whole system is based on ordinary devices such as an analog CCD video camera, a Capsure frame grabber PCMCIA card by IREZ able to convert images with a resolution of 320x240 pixels at a rate of up to 30 frame/sec and a Macintosh PowerBook G3-500Mhz.

In the following I'll use this terminology: *pane,* i.e. the *defined* sub-area that can be placed everywhere in the capture video camera area with whatever dimensions; *frame,* i.e. the *detected* rectangle that delimits the shape of one hand considered as the reconstructed white spot in memory, therefore defined as *spot-hand.*

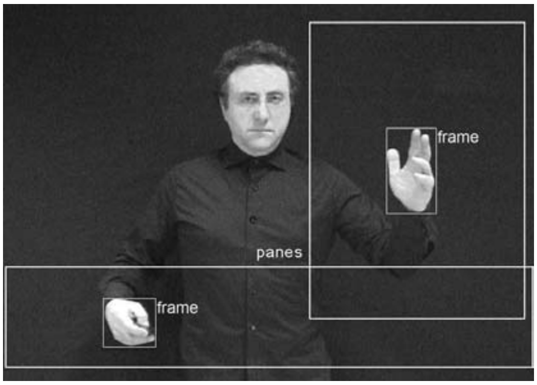


Fig. 2. Typical operative situation

The algorithm that scans and analyses the postures and movements of the hands is simple in principle but, at the same time, it allows a great variety of dynamic figurations truly important for the overall impact on the audience during the performance. In fact it's possible to invent many and new postures and movements to be used in different musical compositions with any sort of free linkage with the theme and the poetics of the music.

I mean that the great variety of shapes, postures and movements of the hands that can be invented by the composer/performer creativity, can be mapped into the *frame* classes so far described.

3.1 Implementation

Handel has been implemented on a laptop PowerBook Macintosh G3 running at 500Mhz so that the system consisting of CCDcamera +CapsureCard +PowerBook can be considered as a generator of information under the control of the performer's hands gesture. That is, Handel can be considered as a general purpose controller which issues messages of the same type of Midi messages issued by, for example, the old MIDI mixer KAWAI MM-16 used for feeding real time musical commercial products such as MAX.

Actually, I use the data streaming for compositions written in pCM (pureCMusic) I realized and presented in the last years in many conferences and meetings [18,19]. The pCM programming framework gives the possibility to write a piece of music in terms of an algorithmic-composition-based program and of synthesis algorithms also controlled by data streaming from external controllers. Everything is written following the C language syntax, compiled into machine code that runs at CPU speed.

The framework consists of a number of functions for sound processing, for generating complex events and for managing external data coming from standard Midi controllers and/or other special gesture interfaces. For Handel I chose to use the UDP protocol because it has two main advantages: it is faster and makes use of a single small flat cable which plugs directly into the laptops outlets so avoiding the presence of two Midi interface-boxes.

3.2 Analysis

Once a frame is grabbed and converted into a matrix of pixels and stored onto memory, the core of the callback routine which implements the functionalities of Handel is invoked. This routine scans the *panes*, search for the spot-hands and, if any, computes and reports dimensions and positions of the *frames*. The *panes* can be dynamically defined by the pCM program/composition as required for different planned musical situations and transmitted to Handel via UDP protocol.

More precisely the callback routine executes the following tasks for each active pane: states the presence/absence of the spot-hand inside pane and, if present, computes the related frame dimensions by scanning the whole pane and storing the highest, the lowest, the leftmost and the rightmost pixels coordinates belonging to the spot-hand.

It's also possible, for each pane, to state the step to be used during the scanning: a step value equal 1 means that every pixel is tested; step 2 means that 1/4 out of the totality of the pixels in the pane are tested; step 3 lowers to 1/8 and so on. The step so defines the grid.

As a consequence the algorithm is faster but, on the other side, does not guarantee all the boundary pixels of the spot-hand are tested: the extreme pixels that delimit the frame dimensions can lay on those rows and columns not tested. In this case a wrong value is issued. This error, however, cannot be greater than the value of the step itself and, considering the great advantage gained in terms of velocity, it results quite acceptable. And in any case the user can freely state that.

Since, actually, very often a gesture controlled performance works on thresholds (for example something has to happen when the mass of a spot-hand becomes greater than a predefined value) to lose precision it's tolerable especially when it's a matter of gaining something crucial such as a true real time control.

The formulas and the operative code program, which put them at work, are the well-known formulas for computing the center of mass previously seen. The frame dimensions are simply given by the difference of the coordinates between the extreme points of the spot-hands.

Usually the hands assume postures that show the palm or the back in respect to the CCDcamera and the audience such as those reported in Fig.2. Fingers can be kept closed together or kept in the fist position. Furthermore many combinations of finger-closed/finger-open (such as when counting) can be taken into consideration. With this class of postures the resulting frames are nearly squares so that values to consider are those related to the mass and its position within the pane.

3.3 Angle of Rotation

A second class of posture produce *flat* frames, i.e. where one dimension is considerably lower in respect to the other. This is the case where the forearm is placed horizontally and the open fingers point to the camera (mime an airplane flight with thumb and little-finger as the wings) or in the posture used in the military salute.

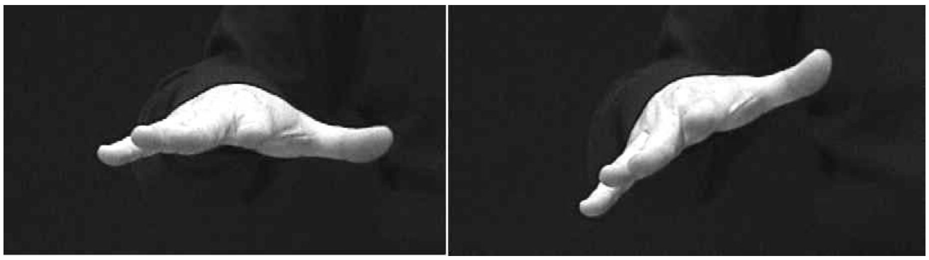


Fig. 3. Flat posture A

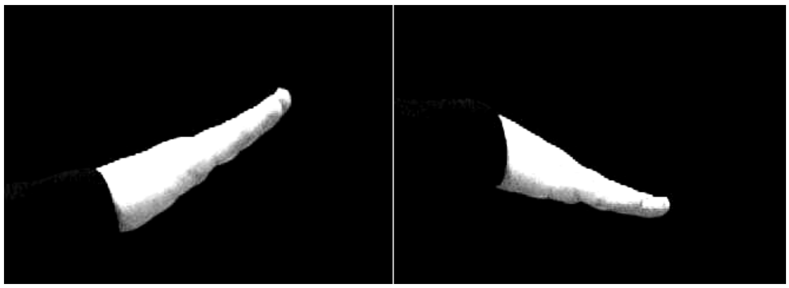


Fig. 4. Flat posture B

With this class of postures the resulting frames are flat and then it makes sense to recognize the angle of rotation. This is computed using the well-known regression-line formulas where x_i and y_i are the coordinates of the white points of the reconstructed spot-hand:

$$slope = m = \frac{S_{xy}}{S_{xx}} \text{ with } S_{xy} = \Sigma\ x_i y_i - \left(\frac{\Sigma x_i \Sigma y_i}{N}\right) \text{ and } S_{xx} = \Sigma\ (x_i)^2 - \frac{(\Sigma\ y_i)^2}{N}$$

As a final remark, I want to recall that it's not a matter of recognizing the shape of the hands as seen in the first approach but, rather, that of freely controlling size, position and rotation of the spot-hands which, in turn, change dimensions and rotations of the frames. At the end the hands really control parametric values for giving expression to real time synthesized music.

As summary, these are the information detected by the program.

- spot-hand **presence** in the pane (true/false).
- spot-hand **barycenter** *(x,y)* coordinates
- spot-hand **frame** dimensions *(base,height)*
- spot-hand **angle** of rotation (if meaningful)

and sent via UDP protocol to the computer that runs the pCM program/composition. The number of active panes defined by the pCM program/composition can be greater than 2 even if, obviously, those really fully controllable at the same time are only two.

3.4 Future Plans

Hardware at the moment in use (PowerBook+Capsure) works very fine. However it is based on obsolete technology: IREZ has not developed the proper drivers for System OS-X so that, should my old PowerBook or the Capsure card go out of order, Handel will be no more usable.

I tried to use webcams based on USB and Firewire protocols but, unfortunately, despite simpler to use, they work with a latency definitely unacceptable in real time applications such as a gesture controlled computer music performance.

For that I'm planning to use the PC-104 standard SBC serie that makes it possible to assembly a very compact special purpose hardware able to: -grab images from analog cameras; -analyse the spot-hands and produce the related values and -guarantee the requested data rate transmission via USB, Firewire and UDP protocols.

4 Conclusions

Apart the way the whole mechanism works and apart the effective usability of this approach to gesture recognition, I found myself to face the problem concerning the performance visual aspect of *gesturing in the air* in front of the audience. Traditional musical instruments force musicians to assume precise postures of the body and specific movements of the hands in relationship with their mechanic and physical acoustic characteristics.

As a novelty, Handel proposes something where *who* controls and *what* is controlled overlap: the hands are at the same time the instrument and the player. Here, no real instrument exists that forces the performer to specific postures and gestures. Therefore, to be completely free induces to search for a new coherence and elegance to take into account while performing.

I found a first answer to this problem by observing gestures of magicians. In fact, very often -if not always- after my concerts played using the TwinTowers and the ImaginaryPiano, people from the audience freely report to me their impression of having watched a magician beside a musician (the italian word for magician is *prestigiatore* actually a contraction of *presto-digitatore* which means "he who moves fingers quickly"). However, I was not quite satisfied with it because in my performances there is no trick or cheating.

Where I found a deep and valid answer is in *Tai Chi Chuan* that has a considerable variety of movements and postures and, as an Oriental Art, helped me to gain awareness about *unity* (yoga) between body and mind. I want to highlight that I'm not going to try an artistic and/or poetic linkage between the two disciplines just because Tai Chi is not *show* but individual and spiritual research. In any case I feel myself legitimated to use what I learned from Tai Chi about control and coordination of my hands.

5 Acknowledgements

I want to thank Massimo Magrini who, as a very skilled and talented musician and expert in acoustics, electronics and informatics, helped me to choose the necessary hardware and to implement the low level crucial routines for grabbing images into memory and for data transmission via UDP protocol. Also thanks to Ubaldo Swami, my Tai-Chi teacher.

References

1. Bordegoni M., Faconti G.P.: Architectural Models of Gesture System, in P.A.Harling and A. Edwards, editors, Proceedings of Gesture Workshop '96, Springer Verlag, pp.61-74, 1996.
2. Rubine D.: Specifying gesture by example, in Computer Graphics, V.25(4), pp.329-337. ACM Press, 1991.
3. Hartung K., Münch S., Schomaker L., Guiard-Marigny T., Le Goff B., MacLaverty R., Nijtmans J., Camurri A., Defée I., Benoît C.: Development of a System Architecture for the Acquisition, Integration, and Representation of Multimodal Information, A Report of the ESPRIT Project 8579 MIAMI, -WP 3-, March 1996.
4. Duke D.J.: Reasoning about gestural interaction, in Proceedings of Eurographics '95, NCC Blackwellm, pp 55-56, 1995,

5. Nagashima Y.: Interactive multimedia performance with bio-sensing and bio-feedback, Shizuoka University of Art and Culture & Science Laboratory, in Int. Conference on Auditory Display, 1794-1 Nuguchi, Hamamatsu, Shizuoka 430-8533, JAPAN, 2002.
6. Wexelblat A.: An approach to natural gesture in virtual environment, in ACM ToCHI, ACM Press, pp.179-200, 1996.
7. Paradiso, J.: Electronic Music: New Ways to Play, IEEE Spectrum Computer Society Press. pp. 18-30, Dec. 1997.
8. Rowe, R.: Machine Musicianship. Cambridge: MIT Press. March 2001, pp. 343-353, 2001.
9. Bertini G., Carosi P.: Light Baton, A System for Conducting Computer Music Performance, "Interface" (Journal of New Music Research), Lisse, Netherlands, Vol 22 N° 3, pp. 243-247, 1993.
10. Tarabella, L., Magrini M., Scapellato G.: Devices for interactive computer music and computer graphics performances: In Procs of IEEE First Workshop on Multimedia Signal Processing, Princeton, NJ, USA - IEEE cat.n.97TH8256, 1997.
11. New Music for Musical Expression, NIME-02, Dublin, Ireland, May 2002.
12. Cardini M., Tarabella L.: Wireless, Premio Marconi 2002 per l'arte tecnologica, Circolo Artistico e Università di Bologna, Corte Isolani 7/a, Bologna, 2002.
13. Cardini M.: Segno elettronico contemporaneo, International Conference on “Tecnologie e forme nell'arte e nella scienza”, Università degli studi di Salerno, 2003
14. Tarabella,L., Bertini G.: About the Role of Mapping in gesture controlled live computer music, in Proceedings of the International Symposium on Computer Music Modeling and Retrieval (CMMR 2003), (Montpellier, France), Lecture Notes in Computer Science (LNCS 2771), Springer Verlag, pp. 217-224, May 2003.
15. Cappellini V.: Elaborazione numerica delle immagini, Editore Boringhieri SpA, Torino, 1985.
16. Tarabella L., Magrini M., Scapellato G.: A system for recognizing shape, position and rotation of the hands, in Proceedings of th Internationl Computer Music Conference ’98 pp 288-291, ICMA S.Francisco, 1998.
17. Povall R.: Realtime control of audio and video through physical motion: Steim's Bigeye: In Proc. Journées d'Informatique Musicale, 1996.
18. Tarabella L.: The pCM framework for realtime sound and music generation , in Proceedings of the XIV Colloquium on Musical Informatics, Firenze, Italy, May 2003.
19. Tarabella L.: TheObject-Oriented pureCMusic framework, in Proceedings of the International Conference on Understanding and Creating Music, Seconda Università di Napoli, Facoltà di Matematica, Caserta, 2003.

Open and Closed Form in Interactive Music

Lars Graugaard

lars@graugaard-music.dk

Abstract. Performing music includes substantial listening skills on part of the performer. Performing with an interactive computer requires the performer to interact with the computer and intuitively and consciously include this information in the responsiveness of his playing. The interaction can be expanded to include the performer's high-level decisions typical of open-form notation. These decisions can be used for defining and re-defining the computer's role in the further development of the piece. In this paper I describe how such an open-form notation is used in the interactive man/machine performance environment of my composition 'GUITAR' for acoustic guitar and interactive computer. The performance environment functions as a perception-based multi-parameter space where the performer's score provides means for exploring the space. The open-form notation emphazises the interactive functionality of the space, and a performance becomes one of many possible explorations of the space.

1. Premises of the Composition

1.1. Open Form

Open form in composition has been widely used after 1950 by composers following first experiments of Karlheinz Stockhausen, John Cage, Morton Feldman, Earle Brown and others. Concepts later related to open-form notation have existed since the emergence of notation itself. Well-known composers including Josef Haydn and Wolfgang Amadeus Mozart have composed in open form, albeit quite informally, such as the latter's aleatoric 'Musikalischer Wurfelspiel'. Some classical form schemes such as 'rondo' have been shown to posess open form qualities [Adorno,1970], and characteristics of open form have been shown to exist in more recent closed form compositions such as Claude Debussy's later compositions. Today, open-form notation has different implementations ranging from open subsets inside a closed structure, to open-form notation governing the entire formal layout. On a micro-level notation has been extended to give the performer choices concerning phrasing as well as of pitch and rhythm. Notation in an open form of the basic elements themselves exists in a variety of ways, from fixed notation to graphic notation and combinations of these. Comparison of compositions of extreme indeterminacy by John Cage and extreme serialism by Pierre Boulez has shown that open and closed form can posses similar cognitive characteristics.

An exact definition of open form as opposed to closed form does not exist. Open and closed form can therefore be understood as plausible and non-contradictive notations that can exist within the same composition. This is fundamental to the use of open and closed form within 'GUITAR', where needs for man/machine interaction govern the choice of appropriate notation.

1.2. Open Form and Interaction

Interaction is a functionality of a system, where data continuously is being passed between two components – in this case a performer and a computer – with consequential changes to detail and form. In a performer/computer loop the performer takes into account the perceptual information he receives from his own playing and from the computer on the small-scale level. He uses this information to make decisions on small-scale level, and on medium-scale level in choices of phrasing, pitch and rhythm. By doing so the performer passes further information on to the computer, which in turn is taken into account at subsequent steps. Each step is potentially so small that the interaction can be perceived as continuous, and the performer integrates his performance with the computer's response through perception requirements implied by the score. In a musical and narrative or conversational context, open-form notation becomes a logical consequence of interaction because it switches the focus from the object of perception to the process of perception [DeLio, 1984].

On the small-scale level a performer perceives and balances the timing, phrasing and dynamics of his playing in order for the composition to develop satisfactory in its details and overall form. Open-form notation provides an enlarged framework for such expressive action/reaction. I have taken this concept of interaction a step further permitting the computer component to respond to medium-scale decisions taken by the performer at nodes and points of bifurcation, and to interfere with large-scale formal structures. Furthermore, the interaction is made to allow for cross-relation between all three basic levels of interaction:

- small-scale level of detailed, expressive actions such as timbre, phrasing and timing,
- medium-scale actions such as bifurcation decisions in the open-form section, or
- large-scale formal level where a computer accompaniment in a subsequent section is based on key performance data gathered in a preceding section.

1.3. Further Development in the Case of 'GUITAR'

In accordance with these man/machine interactions and decisions the open form of 'GUITAR' takes on a particular form in performance. The computer decisions are based on analysis of the performance and on the rules and guide-lines laid down in the programming. Actions by the performer therefore influence the computer component in its decision-making on the formal level as well as on a one-to-one relation of performance expression enhancement. The result is an interaction where the three levels of interaction can freely interact.

It has become my understanding while composing and programming ‘GUITAR’ that the performer best can be seen as moving around in a performance space while performing the score. This has in turn influenced my attitude towards composing and appreciating interactive music since time-based concepts of form must align with semi-static multi-dimensional concepts of ‘object’ or ‘state’. This amalgam of attitudes towards composing – or sound-organizing – must take into account a cognition not readily explained by traditional or contemporary composition models. This seem to be a very fertile area for further practical study and theoretical thinking, since it essentially combines two very different approaches to organizing time, without suspending or negating the existing body of work in either area.

2. The Score

2.1. Overall Layout

‘GUITAR’ is divided into three main sections and one transitory section (fig. 1). The first and third section is in closed form, while the second section is in open form. The transitory section is a combination of these notation-forms, as its function is to bridge between the second and third section. First and third section establish, relate and develop key form identifiers of the composition, and they function as a defining frame around the open-form second section and the transition back to closed-form notation.

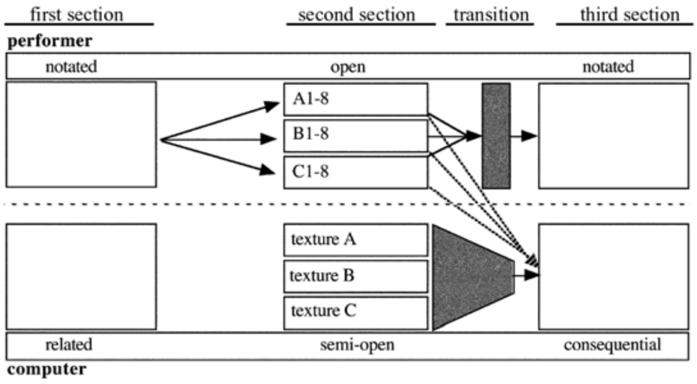


Fig.1. *Overall layout*

2.2. Formal Development

Identity relations (fig. 2, 6, 7) and form identifiers and their development (fig. 3, 4, 5, 6, 7) in sections one and three create perceptual cross-relations and regular reinterpretations. They draw attention to differences and similarities between the performer and the computer. When the material from the first section reappears in the third section, the corresponding changes in the computer component are emphasized.

This new relation in turn points to consequences drawn from the musical ongoings in the second section, and in the transitory section. The time-scale for these activities are on the macro level, typically 1-10 minutes.

Fig. 2.

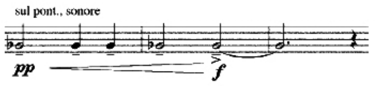


Fig. 3.

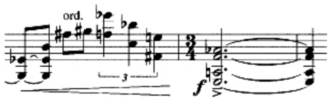


Fig. 4.

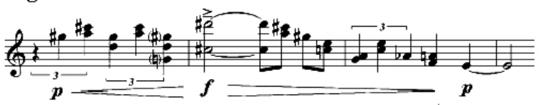


Fig. 5.

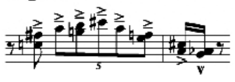


Fig. 6.

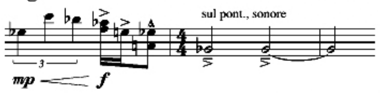
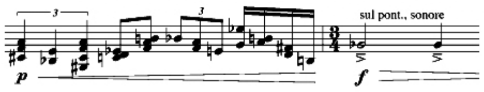


Fig. 7.



The first section develops into an unbalanced and energetic state. This prepares for the second section, and is an example of an unstable and ambiguous form concept common to open form applied to a closed form.

The second section is an open-form notation of three subsections A, B, and C each consisting of eight statements (fig. 8). The statements in each of these subsections are to be played in the order decided by the performer. The only rule given is that in any given subsection the next line in order must be played. The material in the three subsections converge towards the end (fig. 9), but the way and speed with which they converge are decided by the performer.

When the performer has played all 24 statements he continues to the transitory section which is chaotic and random as a result of the compression subsections A, B, and C have undergone. The transitory section ends with a sequence of exclamations after which section three begins. The gradual progression from calm order in the beginning of the composition into complete chaos reached at this point remains unresolved. Instead, section three which contains numerous cross-relations to section one begins.

Fig. 8. *Subsections A, B and C; first statements*

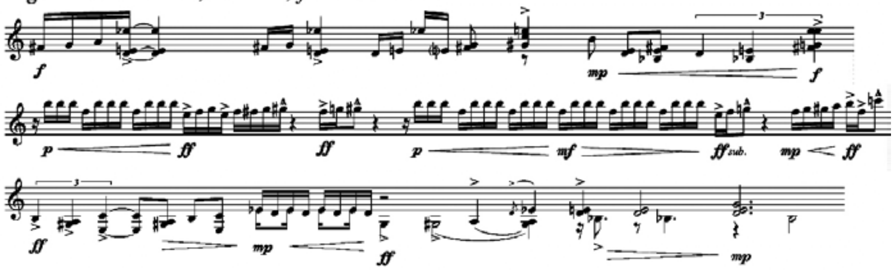
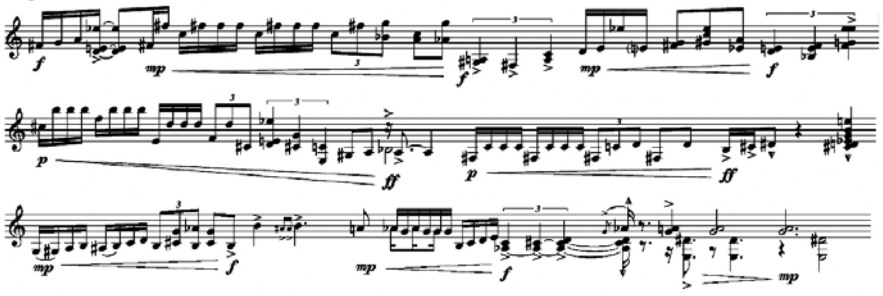


Fig. 9. *Subsections A, B and C; last statements*



2.3. Score-Computer Relations

Entities, moments or passages in the score lets the performer pass on musical expression information in real-time to the computer. This information is used by the computer to influence the synthesis and generative algorithm parameters. The computer may use this information for immediate processing or it may be stored for later use.

When the computer response is perceived to react on the small and medium time-scale of 0-5 seconds it will have consequences for the performer on the level of musical interpretation with consequences for phrasing and open form choices provided by the score. It provides the performer with a degree of expressive control over the computer while the performer in turn also is influenced by the computer response. When the response lies on a larger time-scale it will have consequences for the overall formal development of the piece. However, it is not possible to make a complete distinction between the influence on these time-scales since choices in the open form section are influenced by the sum influences exercised on the small time-scale.

3. The Computer Component

3.1. Software Environment

The computer part is realized using the max/msp software package [Zicarelli, 2004]. Objects used are kernel objects, external objects, and abstractions. A number of algorithms in the form of 'patches' are used.

3.2. Analysis

The acoustic guitar signal is picked up by a microphone before it is digitized and analyzed. Pitch estimation and spectral information is extracted by means of an FFT algorithm in an external object called *fiddle~* developed by Miller Puckette [Puckette, 1998]. Amplitude following such as continuous envelope extraction and attack, threshold and rest detection with derived note duration information is also taking place.

3.3. Signal Transformation

Signal transformation methods used are waveshaping, harmonizing, filtering, delay through multiple variable delays, amplitude modeling, reverb and phase-based spatialization.

3.4. Synthesis

Synthesis methods used are modulation of frequency and amplitude.

3.5. Sampling

A granulation technique using real-time and pre-recorded audio material is employed. The grain algorithm is based on the work of Cort Lippe [Lippe, 1994]. Spatialization of the grains is made according to ideas derived from statistic models.

3.6. Digital Signal Processing Chain

All of the above mentioned DSP techniques can be routed to become input of any other technique provided that technique accepts a signal. The techniques are not all running simultaneously as it would task the cpu too much. Configurations are created on the fly as needed and processing techniques are turned on and off accordingly.

4. The Performer/Computer Relation in Performance

4.1. Analysis Considerations

Performance data extraction of an acoustic guitar is not straight-forward. Polyphonic playing and resonance from strings affect the reading of the spectrum and make for ambiguous analysis results. Playing artifacts from string depression on the fretboard and noise from the wound strings when playing positions are changed produce noise and irrelevant pitch information. Some of these issues can be addressed by treatment of analog signal prior to analysis and subsequent treatment of the extracted data. It may influence choices in the composition at points where unambiguous signal reading is impossible to achieve.

I have made this lack of precise information a characteristic towards the end of the second section in 'GUITAR'. To successfully do so I had to consider it inside the concept of the composition because the use of such ambiguity must be in good consequence with the context in which it appears. The focus of attention at the moment of composing shifted the technical limitation to one of compositional approach. The lack of precise information became a legal component of 'GUITAR' since the chaotic end of the open form section continued into the transitory section. This section is a high-point of chaos before a returning to the ordered closed-form notation takes place. The characteristics of the instrument, its performance, and of the analysis method thereby came to have important influence on the composition, further emphasizing the concept of an interwoven man/machine texture.

4.2. Some Characteristics of Interaction

Real-time DSP parameter manipulation readily relates to the perceptual nature of performance skills and is an effective interaction technique. A performer makes instant physical adjustments for to control his sound based on the feedback he gets from a combination of his auditory and tactile senses. This feedback is combined with the feedback from the computer through the loudspeakers. DSP parameter manipulation tied to performance analysis is part of the feedback to the performer's combined auditory and tactile senses in as far as he feels such a connection to exist. If he feels he influences or controls the computer sound he will seek to integrate it in his expressive performance.

I therefore make a distinction between interaction and triggering which I see as being at the extremes of a man/machine relation. An algorithm may be triggered to begin generating data without any further interaction between performance data and the data generated. A soundfile may be triggered for playback without any interaction between the soundfile – or any point in the chain between the soundfile and its conversion to analog audio – and performance data. These actions are pure triggering. On the other extreme interaction can be in the form of the previously described continuous, real-time affect of digital synthesis algorithms, of algorithms generating such data, or through affecting the behavior of high-level compositional algorithms such as those used for computer decisions on the form level. These actions are interactive. But mostly a combination of computer generated and performance derived data is active in some form and it is seldom so that pure triggering or pure interaction his taking place. The combination and oscillating betweeen mostly triggering and mostly interaction is a way to provide integrity to the compositional ideas of the author while giving the performer a true sense of expressive and interpretive influence on the man/machine system.

4.3. Interaction in the Case of 'GUITAR'

In the first section the computer is responding to the performance with standard processing techniques. Selected synthesis parameters are affected by performance analysis in order to enhance the expressivity of the computer part. In the second section the computer has three textures to accompany the performance of each of the subsections A, B, and C. The performer decides which of the subsections to perform and the computer identifies the subsection during the first 2-3 seconds by detecting key pitches. The transitory section binds all the computer textures together in a rather chaotic musical expression. It is the dynamic high-point of the composition and relevant processing methods from the subsections are combined, together with elaborate spatialization. The computer part in the third section is influenced by the performance of the subsections in the second part as it decides on the range of samples to be used for granular synthesis in the third section.

An example in 'GUITAR' of real-time parameter manipulation technique based on performance analysis is grain size, transposition, and spatial placement in granular synthesis determined by relative weight of harmonics extracted from the analysis of the guitar. In its lower range the acoustic guitar has a relative greater weight on the 1st harmonic in softer dynamics and greater weight of higher harmonics in louder dynamics. This becomes apparent when a sustained note is analyzed since the spectrum after a few seconds almost only contains energy at the 1st harmonic. By tying grain parameters to this relative balance the displacement is controlled by note attack, note loudness, and note duration. This relationship may not be obvious to the performer at the first rehearsal of the piece. It may become obvious during the rehearsals and in general the performer's understanding of such interaction benefits the interpretation of the piece. The decision to hide or emphasize a relation of this nature is not always obvious as some interaction techniques may yield subtle, albeit important results.

Such interaction on a smaller time-scale may also have consequences on a larger time scale. If it is expression-enhancing data it can be hidden, but it could arguably be emphasized if it affects form or the development of a form unit, such as a section. Since it affects the performance, it affects the intensities of the statements which articulate the formal elements, thereby affecting the balance between these elements and consequently the relative weight of the entire section. In 'GUITAR' this is indeed the case and the performer is requested to take this perceptual information into account when performing the section in question.

Interaction techniques spanning larger time intervals are also employed. In the second section a number of audio samples are taken of the performance. The samples are ordered into three collections A.coll, B.coll, and C.coll, according to during which of the three subsections A, B, or C they were taken. Only one of these collections will be used for granular synthesis in the third section. The decision of which of the three collections to use is based on the relative weight by which the performer performed the three subsections. It is calculated as a ratio between average loudness, note density and overall spectrum of each of the three subsections.

5. Concluding Remarks

Computer-supported performance changes the possibilities for musical performance. A dynamic relationship between a performer and an interacting computer can enrich the musical level of performance. The computer provides a dynamic, multi-parameter performance space and the performer explores this space through the performance of the accompanying score. Conceptually, the parameter space is the 'world' and the score is the 'individual'. Time-based interpretations of such a space through score-performance provides new possibilities for musical performances. These explorations can be extended beyond the formal ideas of a strictly musical-compositional narrative.

6. Bibliography

Adorno, Th. W.: *Aesthetic Theory*, 1970.
DeLio, T.: *Circumscribing the Open Universe*, 1984.
Lippe, C.: *Real-time Granular Sampling*, 1994.
Puckette, M., Apel, T., Zicarelli, D.: *Real-time audio analysis for Pd and MSP*, 1998.
Zicarelli, D.: Cycling74 website at *www.cycling74.com*, 2004.

Collaborative Computer-Aided Parameter Exploration for Music and Animation

Daryl H. Hepting and David Gerhard

Department of Computer Science
University of Regina
Regina, Saskatchewan
S4S 0A2 CANADA
{hepting, gerhard}@cs.uregina.ca

Abstract. Although many artists have worked to create associations between music and animation, this has traditionally be done by developing one to suit the pre-existing other, as in visualization or sonification. The approach we employ in this work is to enable the simultaneous development of both music and sound from a common and rather generic central parameter variation, which may simply indicate a structure for periodic repetitions. This central parameter variation is then simultaneously mapped to appropriate musical and graphical variables by the musician and the animator, thereby contributing their own interpretations. The result of this mapping is then rendered in an intermediate form where music and animation are allowed to iteratively influence each other. The main piece of software in this development is the system which allows exploration of parameter mappings. The software interface allows both musician and animator to meaningfully experiment with the other's mappings since the interface permits access in a common form, without requiring additional skills to interpret.

1 Introduction

Throughout recent history, various artists have explored the relationship between music and colour and between a soundtrack and its animation. While early attempts to generate both sound and music from the same composition met only with mild success, it has been observed by artists like Barry Spinello [26] that "we are now at the point where film and music have gone their separate ways so that the only conciliation of the two seems to be some form of 'synchronization'; that is music will be composed for an existing film sequence or vice versa."

Current examples of this sort of conciliation are the work done by Lytle at *Animusic* [17], where animation is generated to correspond to a pre-existing MIDI score, and the work done by a composer creating the score for a movie once that movie is complete. Popular music visualizers, like WinAmp [22] and iTunes [1] use a different approach that is reactive in nature since they are intended to create a visual for the sound being heard at the moment. This reaction is not dissimilar to the way pianists may have approached the task of accompanying silent films in years past, although repetition for them might have improved their interpretation for subsequent screenings. While Lytle's work creates visualizations for sounds at the note level, WinAmp makes visualizations at the sample level. However, neither allows for any interaction between the sound and the visualization.

Examples abound of either the music or the animation driving the development of the other, either in realtime or not. However, little has been done to explore a process that allows the music and animation to be developed together by mapping each from a single central parameter variation. The resulting intermediate music and animation can further be used to inform each other, to create more interconnections and nuances in the final work. The principle advantage of this parameterization is that musician and animator can both interact with the piece using the same parameter set so that one can affect the portion under the domain of the other. This empowers the animator to say "music like this" or the musician to similarly say "pictures like this."

This paper describes a process that allows musicians and animators to work together and collaborate closely by virtue of computer-support for the parameterizaton and exploration of the resultant parameter spaces. The rest of the paper is organized as follows. Section 2 describes the motivation and prior work. Section 3 describes the present approach and the use of computer-based tools to support it. Section 4 presents an example of how this approach can be used in the development of an integrated work. Section 5 discusses conclusions from our experience and directions for future work.

2 Background

There are different ways to combine music and animation and there are different ways to conceive of the relationship between the two. Schillinger [24] and Whitney [30], to name two, had their own precise ideas about this relationship, which Moritz [20] distinguished as quantitative and qualitative. Bute and Schillinger had an long-lasting collaboration to explore Schillinger's system of music through Bute's animation. The of the previous work can be divided into *visualization* of audio and *sonification* of video.

2.1 Visualization

Audio visualization has been a popular application of cross-modal media generation since computers made it possible to analyze sound in real time. In general, there are two types of audio visualizers: waveform-driven and event-driven. Waveform-driven visualizations encompass the visual plug-ins available in media players such as WinAmp and iTunes. The sound waveform (and often a rudimentary frequency analysis of said waveform) is made available to a plug-in program which renders visual data based on the waveform data. Usually, this consists of drawing and manipulating the actual waveform or spectrum, in the style of an oscilloscope or spectrum analyser. and adding colours and transitions that are unrelated to the sound. This method is consistently successful in generating visual content related to the sound, however it can only react—the visual medium depends entirely on the acoustic waveform in the current time frame, or more commonly of an immediately past time frame. This method can be extended to predict future events, but little work has been done on this, primarily because the iTunes and WinAmp visualizations are doing precisely what they were designed to do: add some visual effects to what is primarily an audio experience. These visualizations are incapable of producing anything contextually relevant, for example a picture of the musician. Their role is strictly augmentative. Even if it were part of the goal to produce a contextually *relevant* visualization, it is very difficult or even impossible to choose the extra visualization information (colour, movement etc) based on acoustic constructs because the available acoustic information is strictly limited to instant realtime waveform and spectrum information.

It should be noted that waveform driven realtime visualizations like iTunes and WinAmp can be augmented with contextual information from ID3 tags, which contain artist, album and year information, as well as *genre* information. Visualizer components can be developed to make use of this information, using specific colour palette for specific genres of music, however this data is limited to the accuracy of the person who entered these tags, and is a single characteristic, constant for the entire sound piece. If a piece of music changes from minor key to major key half way through, this change is not reflected in the ID3 tags, nor is it possible to reflect any structural information like this in the visualization.

The second class of audio visualization is one step removed from the actual waveform, to the *events* that cause the waveform. In this case, a set of paramaters (the list of events) is used to generate both the video and audio media. The classic example of this is Wayne Lytle's Animusic pieces [17, 16]. In this work, a MIDI (musical instrument digital interface) file is used to generate music, and an animation is also generated which corresponds to the music. Virtual musical instruments are designed which actuate according to the MIDI file, so that it appears that the virtual musical instruments are "playing" the music.

The Animusic work is very cohesive because it is animating the instruments or the likely sources of the acoustic events. It works because the domain is auditory and visual at the same time, in the same way that a moving mouth on a face tends to fuse with an auditory speech waveform into an apparant single entity.

Event-based visualizers like Animusic are inherently non-realtime because in many cases, visual events must *precede* the corresponding acoustic events. In order for a visual drum hit to appear to synchronize, or fuse, with the corresponding acoustic event, the drum stick must begin the motion before the event, so that the stick strike will occur at the same time as the sound. Another limitation with this technique is that visualizations can correspond only to acoustic events. An acoustic event without an apparent visual analogue, such as background ambience or continuo, must be visualized in an abstract way and this does not always fuse as well as directly realizable instruments like drums. This abstract visualization, combined with the exaggeration of non-moving instruments like horns, may create some dissonance in the synchronization.

The fundamental limitation of this technique, as well as the realtime visualization mentioned above, in the context of this paper, is that both are inherently unidirectional. All relevant generated visuals are based entirely on the pre-recorded sound. In the best case, the realtime visualization technique can be applied to realtime sound input, and perhaps a person could play an instrument, watch the corresponding visualization, and change the way they play based on the visuals immediately past. Even in this case, all visuals are generated based entirely on the sound waveform, and as such, no *interaction* is possible.

2.2 Sonification

Sonification is a complementary technique, where sound is generated based on numerical or visual parameters [28, 14]. Historically, sonification (also called auralization) has referred to the production of sound based on some data-driven parameters, as an augmentation to data visualization techniques. Sonification of visual representations has been done, but in a very formalized way. A straightforward historical example is the musical score: commercial systems exist which will scan a piece of sheet music and translate the resulting image into a musical representation such as MIDI or MusicXML [8], for playback or manipulation in a music editing program. A more general application of the sonification of graphical objects is composition by time-frequency diagrams, where a composer "draws" the score on a piece of paper in a time-frequency representation, and the computer scans the sheet and produces sound corresponding to the lines drawn by the composer. This is much more versatile than simple score scanning, since dynamic time-frequency constructs such as sweeps and noise clouds are available to the composer, however this method is not realtime and requires a development–evaluation cycle where a figure is drawn first, and after the figure is complete the new composition can be heard.

2.3 Combined Visualization and Sonification

Cross-modal system have been proposed in the past which attempt to integrate visual and acoustic events into a contiguous whole. Of particular note is the work of Hahn et al. [9, 29], where visual and acoustic events are generated together. Their approach seeks to synchronize the development of sound and *motion* by connecting the sound and motion parameters before rendering. Their work is primarily motivated by the addition of sound effects to graphic objects, such as wind-chimes or a spinning disk. The animation of the movement of the object is described using a set of parameters, and these parameters are used to synchronize and generate the associated sound.

In many ways, this is the inverse of Lytle's event-driven music visualization described above. In this case, *visual* events are described in a parameter space and the corresponding *acoustic* objects are generated from this parameter space. The same limitations apply to this method as did to Lytle's *Animusic*: the system is inherently unidirectional and the sounds match the motion only inasmuch as the motion can be seen to generate the sound. Motions with no natural associated sounds cannot be linked to a sound by this method in a unified way.

This should not be taken as a criticism of these methods—*Animusic* and Hahn's integrated approach are both exceptional pieces of work, for their specified domains. The automatic generation of sound effects (also called digital foley) is an interesting and difficult research domain in its own right. Rather, we see to show how our system of interaction is different from these previous systems. It is bidirectional rather than unidirectional, and iterative rather than single-pass. Each medium has the potential to be equally important in the final generation, and what is perhaps most important, the structure of the proposed system is not specific to the media being generated. The system could be expanded to any number of related media, all generated from the central abstract parameter set and augmented by interactions with the other media.

Although the translation of data from one medium to another is a well-established research area, in the above examples, and indeed in much of the related work, one medium has a strictly defined format from which parameters are extracted to generate the second medium. Little has been done in the way of the simultaneous, bidirectional generation of visual and acoustic objects from a common set of parameters. The next section describes a computer-supported approach to facilitate such simultaneous generation.

3 Approach

Parameterizations of music and visual art are available in a variety of forms. Such abstractions are useful to enable a richer exploration of the subject material by broadening one's concept of the material. For example, Stockhausen's music explored different parameters [27]. Boden [5] discusses conceptual spaces defined by stylistic conventions, and how to expand those spaces with three rules: "drop a constraint", "consider the negative", and "vary the variable." The card deck, known as *Oblique Strategies* [6] also attempted to provide a means to acheive such an expansion.

However, the conceptual spaces might not need so much expansion as more effective exploration. In discussing his work for the film *Arabesque*, Whitney says (in 1980) that "this program concept was not deeply 'mined' for its fullest possibilities. . . . It is significant to realize how meager my experience was in exploring the *Arabesque* material, and how limited all exploration is to date, in this area of computer graphics."

Whitney used a *manual* approach, according to Kochhar et al. [15] with respect to the computer. While the computer provided the raw capabilities, Whitney was responsible for putting everything together. Other relationships between human and computer identified by Kochhar et al. [15] are *automatic*, at the other end of the spectrum from manual, and *augmented* which seek to balance manual and automatic characteristics to support the user in his or her task, Automatic systems, which generate solutions with a minimum of user input, are precluded from this task because, as Whitney said "art is a matter of 'judgment – not calculation.' " In terms of visual decision making, Bertin's process [4] comprised matrix analysis of the problem (questions are defined); graphic information-processing (answers are discovered); and graphic communication (answers are communicated). For him, it was clear that this work could never be automated because no machine would ever be able to solve this problem of imagination.

Between these extremes are augmented systems that enable computers and humans to work together. In a "good" interface, according to Baecker et al. [3] human and computer "should augment each other to produce a system that is greater than the sum of its parts. Computer and human share responsibilities, each performing the parts of the task that best suit its capabilities. The computer enchances our cognitive and perceptual strengths, and counteracts our weaknesses. People are responsible for the things the machine cannot or should not do." For Baecker et al, [3], human capabilities include judgment and creativity and computer capabilities include patience and reliable memory.

The *cogito* system [10, 11] has been developed as a means to preserve the user's ability to exert judgment while exploring a large, complex parameter space. Points in the space are constructed from all possible combinations of values from the different parameters. For example, given 3 parameters each with 10 possible values, the space has $10^3 = 1000$ points. Foley and Ribarsky [7] wrote that only automatic methods could be used effectively with large parameter spaces because otherwise the users would be overwhelmed. Users of the *cogito* system have not been overwhelmed, however, even though they deal with millions of possible combinations [10]. The system, depicted schematically in Figure 1, allows users to apply their own judgment when assessing alternatives automatically generated by the system, which has the patience to realize these different combinations and keep track of them.

The system functions in two ways: first it allows people to explore different relationships among parameters. By reorganizing the view of the parameter space, as shown in Figure 2, it is possible to transform one's perception of the parameter space from Klondike (difficult to navigate) to homing (easy to navigate) and find a desired solution [23]. Secondly, it records and manages the exploration of space, so that the space can be explored in a systematic way and one can return to previous explorations to examine other options at decision points. In this way, the system is an example of a numerical experimentation environment [13]. Where there are examples of interest, the user may click to select them. The subsequent space of alternatives contains all possibilities consistent with those selections. For each parameter, the list of acceptable values comes from those in the selected exemplars.

The *Design Galleries* concept [19] has some interesting features - namely that it can show alternatives that are good examples of a set of values, however the evaluation function has to be specified *a priori.* The *cogito* system permits results to be partitioned based on qualitative or quantitative differences in values within one or more parameters.

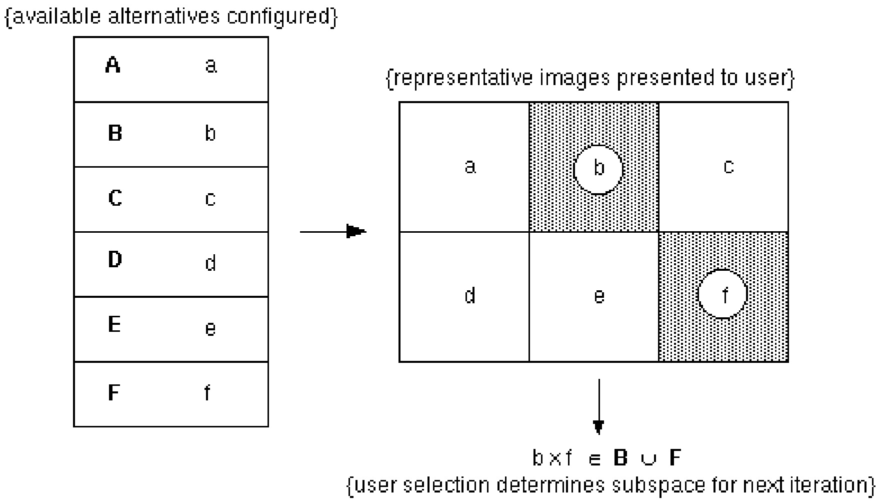

{available alternatives configured}
A a
B b
C c
D d
E e
F f
{representative images presented to user}
a b c
d e f
b×f ∈ B ∪ F
{user selection determines subspace for next iteration}

Fig. 1. Schematic look at the hierarchical interface: the space of available alternatives is grouped according to user-specified criteria. Each group (A – F) has a representative element (a – f) which is displayed to the user. The subspace for the next search iteration is based on the user selection (b and f).

The process of generating media from a parameter space is illustrated in Figure 3. It begins with the selection of a central parameter variation, followed by the selection of a mapping strategy to move from the central parameters to the media. A separate, independent mapping strategy can be selected for each medium in question, in this case for sound and video. Once the mapping strategy has been selected, the media are generated. At this stage, there are three options. The generated media can be taken as complete, the central parameter variations can be re-visited, or a set of *intermediate* parameters can be derived from the generated media. These extracted parameters are then mapped to media-specific constructs, and the media is re-rendered. Each parameter set (central and intermediate) can be re-visited as many times as necessary.

This parameter variation is expressed in generic terms so that both musician and animator have equal access to its modification. Such variations could be constructed in different ways: mathematically in terms of trigonometric functions or visually by sketching curves then copying, scaling, and translating them. Figure 4 illustrates one set of parameters used by Whitney [30]. It shows 12 points and the differential movement applied to each so that at the end of the cycle, the points are once again coincident. This illustrates in general Whitney's ideas about digital harmony, which he described in the context "the relationship of the three terms: **differential**, **resonance**, and **harmony**. First, motion becomes patterns if objects move *differentially.* Second, a resolution to order in patterns of motion occurs at points of *resonance.* And third, this resolution at resonant

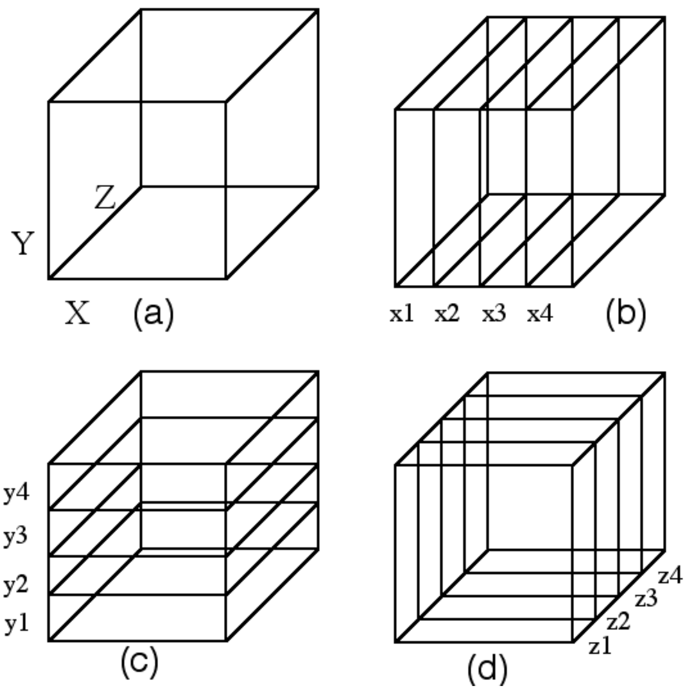


Fig. 2. Consider a three-dimensional space, depicted in the top left, with axes X, Y, and Z. Organizing the space in terms of any of those 3 axes leads to the other states depicted. If elements in component X are chosen sequentially, those in Y and Z can be selected randomly to give a sense available options.

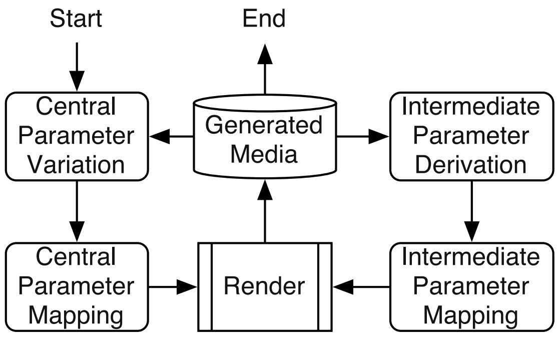

Start
End
Central Parameter Variation
Generated Media
Intermediate Parameter Derivation
Central Parameter Mapping
Render
Intermediate Parameter Mapping

Fig. 3. The process of generating media from a parameter space. Begin by choosing a central parameter variation which can be mapped separately to parameters in the music and animation. Once the media has been generated, intermediate parameters can be extracted and mapped, and the central parameter set can be re-mapped until a desired effect is achieved. At each iteration, the generated media is available.

events, especially whole number ratios, characterizes the differential resonant phenomena of visual *harmony*."

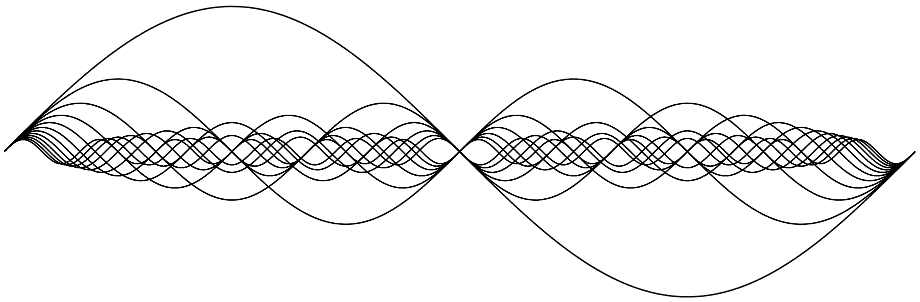


Fig. 4. Parameter variations illustrating Whitney's concept of digital harmony (after Whitney).

This central parameter variation provides what can be considered the *score* of the integrated piece being composed. As such, the central parameter variation is represented with each parameter on a separate line, as in Figure 5. The score analogy is appropriate for musicians but also for animators since it is very similar to features present in most non-linear video editing programs and multimedia authoring systems. Such a representation enables direct access to any point in the work, allowing it to be played. The common and unbiased form of representation helps both musicians and animators contribute to the integrated work, without requiring a great deal of effort on the interface which has a low syntactic burden and seeks to minimize the gulfs of execution and evaluation [21].

When an orchestra plays a piece of music, a score is used to tell each instrument what to do. In the same way, the central parameters can be considered a score for the generation of the sound and animation. Each acoustic and visual event is generated based on the way the parameters change. The first level of media creation, then, can be considered as an orchestra playing a score. Once the first pass of media creation is complete, parameters are extracted from each medium and used to augment the corresponding medium, so extracted visual information (like the area of the figure) is used to augment the audio signal, while extracted audio parameters (like the loudness of the waveform) are used to augment the visual signal. These new parameter tracks are added to the overall score to generate the next level of audio and visual media. This process is closer to the idea of a group of improvisational jazz musicians playing a song together. Each musician has the lead sheet in front of them so that they are all playing the same song, but each individual then can alter or augment their playing based on the interaction with the rest of the musicians. The lead sheet (the score, the central parameters) is fixed, and when used as a starting point, ensures cohesion between the players, but it is not the whole story, and the final music generated by the players is more than the sum of the individual interactions.

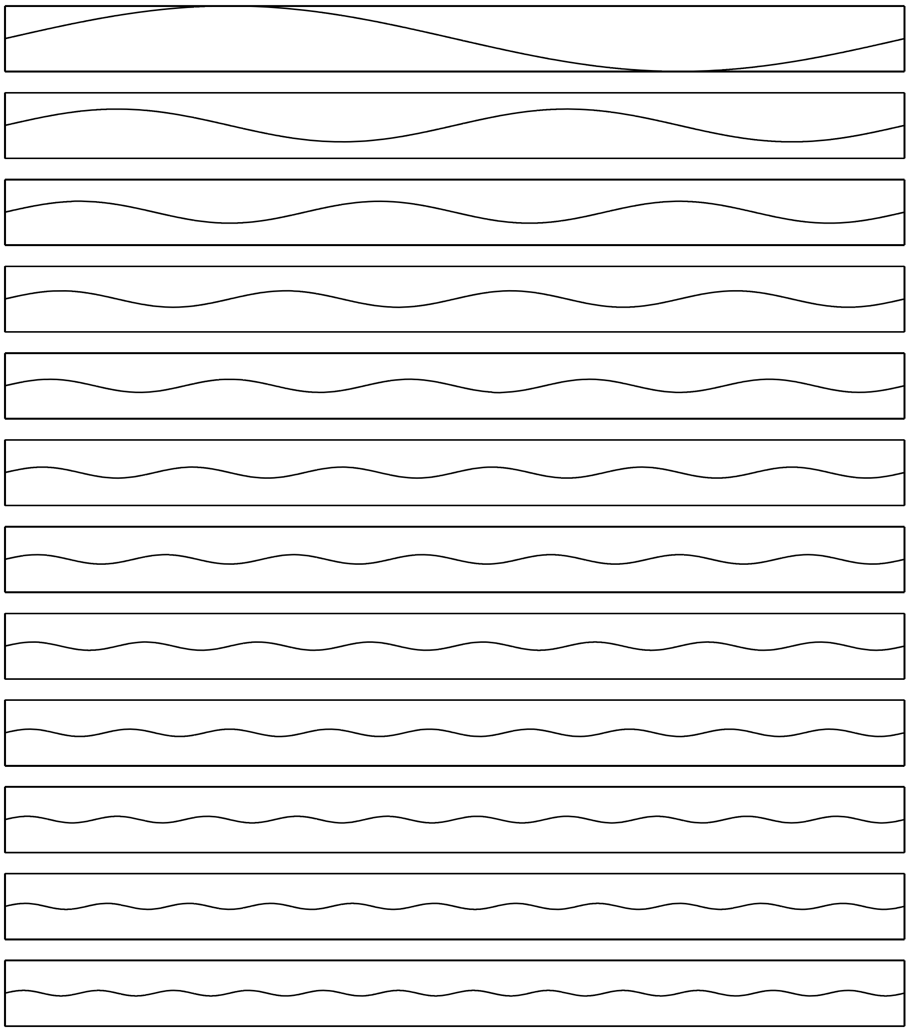


Fig. 5. The Whitney parameter variations from Figure 4 , written in our parameter score format.

The next phase involves the musician and animator each creating mappings from the central parameter variation to their respective media. It could be done in a variety of ways: algorithmically by modulating some facet of the media as a function of a particular central parameter component, or by choosing one or more keyframes and fitting them onto the central parameter variations. These exemplars may come from a variety of sources. The musician may play sounds or phrases on a keyboard, and the artist might draw keyframes to be analysed.

The *cogito* system can be used to present different key visual frames or sounds for evaluation by the user.

Once the mapping has been established, the work can be rendered in a preliminary fashion. At this point, additional parameters can be derived from both the music and the animation. These parameters can be used to influence and add depth to the work in its present preliminary form. These additional parameters can be used either to map continuously onto an effect (such as adding reverb to a sound) or to drive a threshold function which can trigger an associated event (such as adding a musical note event). The *cogito* system could again be used to explore the different applications of the derived parameters within the whole work, serving as the basis for further iteration.

The goal of the *cogito* system is to allow the exploration of a space to find the particular combinations that meet the needs of the user. Simon [25] called these *satisficing* solutions. We get what is desired at a specific point in time, recognizing that what is desired may change over time. It is helpful then to be able to go back and revisit past decisions. In moving towards a piece that is accepted by a wide audience, we may apply constraints from a range of sources, to improve the final product.

4 Example

A short experimental piece, entitled *Triangularhythmic*, was created by the authors with the help of Matthew McKague. The central parameter variation used is based on groups of three. This was chosen with foreknowledge of the visual space that would be explored, and with an idea of the animation mapping that might be used. The visual space is that of fractals [18] in general, and in particular those generated from sets of contractive affine transformations [12]. The sonic space is generated using rhythmic loops designed to fit the fractal transformations at the central parameter level, and composed using Soundtrack [2].

The parameter score for this example appears in Figure 6. The top five parameter tracks correspond to central parameters used to generate the first pass acoustic and visual media. The bottom two parameter tracks are examples of extracted parameters. Figure 7 shows the Fourier spectrum of the generated soundtrack.

There are several benefits of this representation. One can see the whole timeline at once, or isolate a specific segment, and see the *interaction* between parameters. This is a *dynamic* representation, meaning that while the central parameter score remains the same throughout the process, one can add secondary parameters, such as the extracted parameters discussed above, and investigate and understand the interaction between the central parameters and these new extracted parameters. It is not necessary to show or describe the mapping that generates these secondary parameters, since the original central parameters and the new extracted parameters are both present in the representation. The parameter set mapping is inherent in the representation.

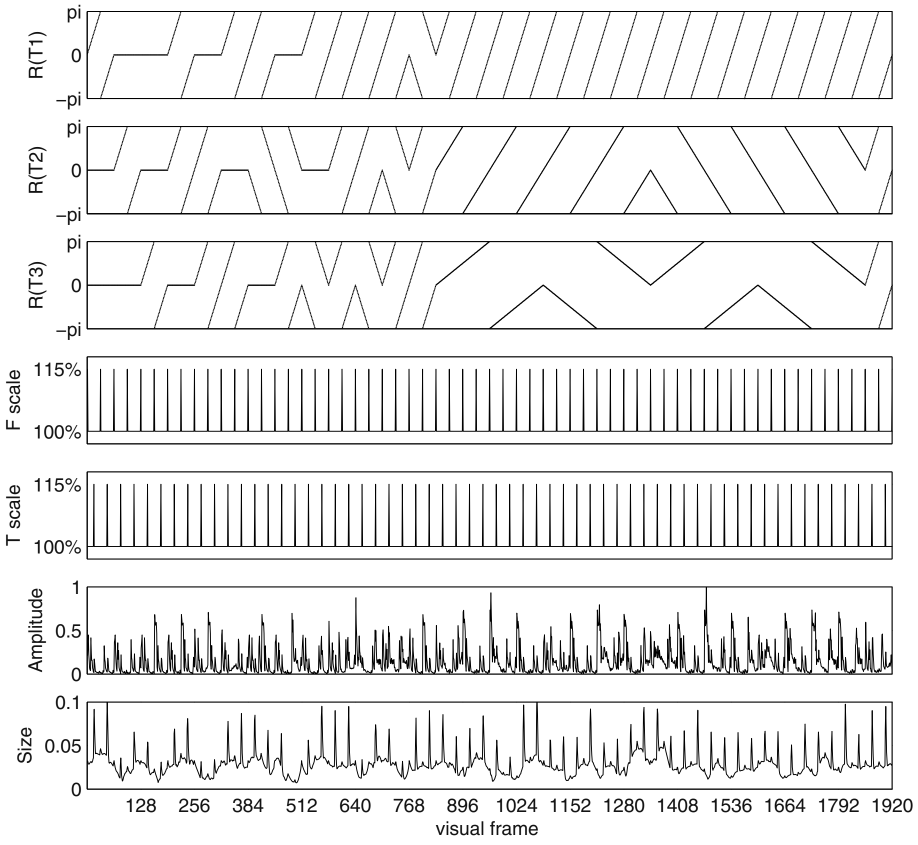


Fig. 6. Score of parameters. R(Tn) is the rotation factor for each transformation. F scale is the scale factor for the entire figure. T scale is the scale factor for the transformations. Amplitude is the power level in each frame of the generated sound file, an extracted parameter. Size is the number of non-black pixels in each frame of the generated animation, an extracted parameter.

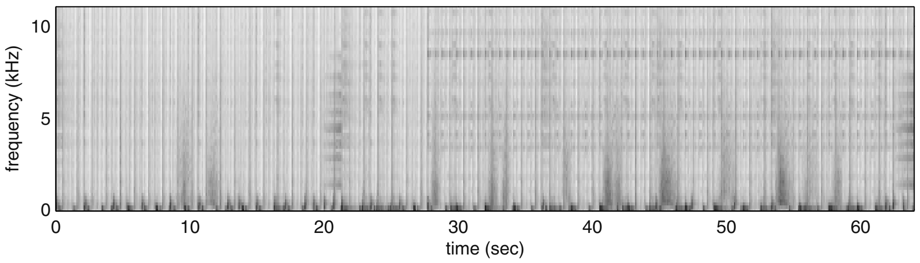


Fig. 7. Fourier spectrum of the sound track generated using the parameter mappings.

The resonance inherent in the central parameter variation has been emphasized in both the music and animation, with the strong beats in the music and the scaling applied to the animation. The beats present in the central parameters also show up as beats in the secondary parameters.

5 Conclusions

The motivation for this work was the development of a process by which a single, central, parameter variation could be used to develop intimately related music and animation. Our initial experience with this approach has been very encouraging. Long-term success with this approach is only attainable with a highly developed computer-support structure, similar to the one presented here, which allows the musician and animator to exercise their judgment while at the same time sparing them from the details of the parameters being explored. In future, we will look at ways to provide musicians and animators with more ready access to the parameterizations and schemas put forward by various artists, Schillinger [24] and Whitney [30] included. Conceptually, Whitney's ideas have meshed well with the approach we have presented here.

The use of parameter spaces by artists is directly related to the control they have over the parameters. The systematic exploration of parameter spaces as described here allows the artist to pass on some of the"grunt work" of navigating the space of alternatives for composition to the computer. The system does not limit the artist's creativity in any way, but it also does not require the artist to examine every permutation and possibility that is contained within the parameter space. This entails a great deal of balance while removing tedium without increasing restrictions. An exhaustive search is impossible, but a fully constrained search is uninteresting, and risks missing desired areas of the parameter space. The artist using the system can therefore discover global areas of interest and iteratively constrain the search space to extract more and more detail, with the consistent option of re-tracing and examining other areas. The goal is to enhance the composition process by enabling exploration based on the artist's conceptualization of the problem, which is translated into an appropriate parameterization of the problem space.

The discovery of appropriate parameter mapping paradigms is another issue which depends on the situation. In some mediums, there exist well known and well used parameterizations, and these should be utilized whenever possible. This allows the artist to continue to use paradigms with which he or she is familiar, and perhaps examine them in a new light, but again, they should not be used to artificially constrain the exploration.

Further work will include the investigation of which parameters artists will use within the context of this type of system, and how fully they are able to make use of the extended abilities such a system could give them. To this end, we will prepare and execute perceptual experiments to ascertain the interaction modes that are most helpful to the collaborative composition process, since part of the goal of this work is to make composition more accessible in general. Interfaces will be developed and examined to free composition from the domain of experienced composers. It should be noted that this does not introduce a limit on virtuosity, and experienced and insightful artists should in no way be threatened by the ability of amateur artists to explore and compose.

6 Acknowledgments

Matthew McKague and Tim Maciag have helped to develop the system to its current state. This work was supported by the University of Regina and the Natural Sciences and Engineering Research Council of Canada.

References

[1] Apple Computer. *iTunes.* Online: [http://www.apple.com/itunes/], Retrieved January 29, 2004.

[2] Apple Computer. *Soundtrack.* Online: [http://www.apple.com/soundtrack/], Retrieved January 29,2004.

[3] R. M. Baecker, J. Grudin, W. A. S. Buxton, and S. Greenberg. *Readings in Human-Computer Interaction: Toward the Year 2000.* Morgan Kaufmann, 1995.

[4] J. Bertin. *Graphics and Graphic Information-Processing.* de Gruyter, 1981. Translated by W. J. Berg and P. Scott.

[5] M. A. Boden. Creativity. In M. A. Boden, editor, *Artificial Intelligence: Handbook of Perception and Cognition*, chapter 9, pages 267–291. Academic Press, 2nd edition, 1996.

[6] B. Eno and P. Schmidt. *Oblique Strategies: One Hundred Worthwhile Dilemmas.* Online: [http://www.rtqe.net/ObliqueStrategies/], Retrieved January 29, 2004.

[7] J. Foley and W. Ribarsky. Next-generation data visualization tools. In L. Rosenblum et al., editor, *Scientific Visualization.* Academic Press, 1994.

[8] Michael Good. MusicXML for notation and analysis. In W.B. Hewlett and E. Selfridge-Field, editors, *The Virtual Score*, pages 113–124. MIT Press, Cambridge, 2001.

[9] J. Hahn, J. Geigel, J. W. Lee, L. Gritz, T. Takala, and S. Mishra. An integrated approach to motion and sound. *Journal of visualization and computer animation*, 6(2):109–123, 1995.

[10] D. H. Hepting. Interactive evolution for systematic exploration of a parameter space. In C. H. Dagli et al., editor, *Intelligent Engineering Systems through Artificial Neural Networks*, volume 13, pages 125–131. ASME Press, 2003.

[11] D. H. Hepting, F. D. Fracchia, J. C. Dill, and R. D. Russell. Cogito: a system for computer-aided visualization. Technical Report CMPT TR 96-02, Simon Fraser University, 1996. Unpublished technical sketch presented at SIGGRAPH 96.

[12] D. H. Hepting, P. Prusinkiewicz, and D. Saupe. Rendering methods for iterated function systems. In H.-O. Peitgen, J. M. Henriques, and L. F. Penedo, editors, *Fractals in the Fundamental and Applied Sciences*, pages 183–224, New York, 1991. North-Holland. Reprinted in *Fractals: from Folk Art to Hyperreality, SIGGRAPH 92 Course #12 Notes.*, Edited by P. Prusinkiewicz. ACM SIGGRAPH, New York, 1992, pp. 3-1 to 3-41.

[13] Y. E. Ioannidis, M. Livny, S. Gupta, and N. Ponnekanti. Zoo: A desktop experiment management environment. In *Proceedings of the 22nd International VLDB Conference*, pages 274–285, 1996.

[14] Joseph Wayand John G. Neuhoff, Gregory Kramer. Sonification and the interaction of perceptual dimensions. In *ICAD*, 2000.

[15] S. Kochhar, J. Marks, and M. Friedell. Interaction paradigms for human-computer cooperation in graphical-object modelling. In S. MacKay and E. M. Kidd, editors, *Proceedings of Graphics Interface '91*, pages 180–189, 1991.

[16] W. Lytle. Animusic: a computer animation video album (DVD), 2001.

[17] Wayne Lytle. *Computer Animated Music*. Online: [http://www.animusic.com/software.html], Retrieved January 29,2004.

[18] B. B. Mandelbrot. *The Fractal Geometry of Nature*. W. H. Freeman, New York, 1982.

[19] J. Marks et al. Design Galleries: A general approach to setting parameters for computer graphics and animation. In *SIGGRAPH '97 Conference Proceedings*, pages 389–400, 1997.

[20] W. Moritz. Mary ellen bute: Seeing sound. *Animation World Magazine*, May 1996.

[21] D. A. Norman. *The Psychology of Everyday Things*. Basic Books, New York, 1988.

[22] Nullsoft. *Winamp*. Online: [http://www.winamp.com/plugins/], Retrieved January 29, 2004.

[23] D. N. Perkins. Insight in minds and genes. In R. J. Sternberg and J. E. Davidson, editors, *The Nature of Insight*, pages 495–533. MIT Press, Cambridge, MA, 1995.

[24] J. Schillinger. *The Schillinger System of Musical Composition*. Carl Fischer, 1946.

[25] H. Simon. *Models of Discovery*. Reidel, 1977.

[26] B. Spinello. Notes on "Soundtrack". *Source - Music of the Avant Garde*, 1970.

[27] K. Stockhausen. *Towards a Cosmic Music*. Element Books, 1989.

[28] Russell Storms. Auditory-visual cross-modal perception. In *ICAD*, 2000.

[29] T. Takala and J. Hahn. Sound rendering. In *Computer Graphics: SIGGRAPH '92 Conference Proceedings*, pages 211–220, 1992.

[30] J. Whitney. *Digital Harmony: On the Complementarity of Music and Visual Art*. Magraw-Hill, 1980.

Comparing Pitch Spelling Algorithms on a Large Corpus of Tonal Music

David Meredith

Centre for Computational Creativity, City University, London,
Northampton Square, London, EC1V 0HB, United Kingdom
dave@titanmusic.com
http://www.titanmusic.com

Abstract. This paper focuses on the problem of constructing a reliable pitch spelling algorithm—that is, an algorithm that computes the correct pitch names (e.g., C♯4, B♭5 etc.) of the notes in a passage of tonal music, when given only the onset-time, MIDI note number and possibly the duration of each note. The author's *ps13* algorithm and the pitch spelling algorithms of Cambouropoulos, Temperley and Longuet-Higgins were run on a corpus of tonal music containing 1.73 million notes. *ps13* spelt significantly more of the notes in this corpus correctly than the other algorithms (99.33% correct). However, Temperley's algorithm spelt significantly more intervals between consecutive notes correctly than the other algorithms (99.45% correct). All the algorithms performed less well on classical music than baroque music. However, *ps13* performed more consistently across the various composers and styles than the other algorithms.

1 Introduction

1.1 The Concept of a Pitch Spelling Algorithm

In this paper, I focus on the problem of constructing a reliable *pitch spelling algorithm*—that is, an algorithm that reliably computes the correct pitch names (e.g., C♯4, B♭5 etc.) of the notes in a passage of tonal music, when given only the onset-time, MIDI note number and possibly the duration of each note in the passage.

There are good practical and scientific reasons for attempting to develop a reliable pitch spelling algorithm. First, until such an algorithm is devised, it will be impossible to construct a reliable *MIDI-to-notation transcription algorithm*—that is, an algorithm that reliably computes a correctly notated score of a passage when given only a MIDI file of the passage as input. Second, existing audio transcription systems generate not notated scores but MIDI-like representations as output [1, 2, 3]. So, if one needs to produce a notated score from a digital audio recording, a MIDI-to-notation transcription algorithm (incorporating a pitch spelling algorithm) is required in addition to an audio transcription system.

Third, knowing the letter-names of the pitch events in a passage can be extremely useful in music information retrieval and musical pattern discovery [4].

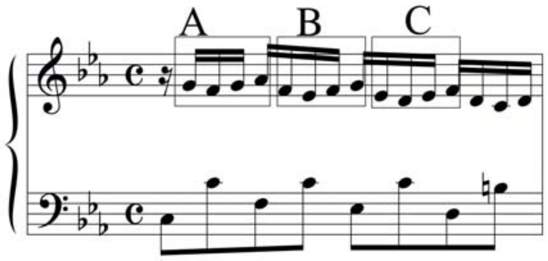


Fig. 1. Three perceptually similar patterns with different chromatic pitch interval structures (from the first bar of the Prelude in C minor (BWV 871/1) from Book 2 of J. S. Bach's *Das Wohltemperirte Clavier*)

In particular, two occurrences of a motive on different degrees of a scale might be perceptually similar even if the corresponding chromatic intervals in the patterns differ. For example, the three patterns A, B and C in Fig. 1 are perceived as being three occurrences of the same motive even though the corresponding chromatic intervals are different in the three patterns. Note that, in this example, one important aspect of the perceived similarity between patterns A, B and C is nicely represented in the notation by the fact that they all have the same scale-step interval structure (i.e., a downward step followed by two consecutive upward steps). In other words, one result of the choice of pitch names for the notes in this passage is that the scale-step interval structures are the same for these three perceptually similar but chromatically different patterns. This illustrates the fact that a correctly notated score is much more than simply a set of instructions for the performer (cf. tablature). A correct Western staff notation score of a passage of tonal music is a structural description of the piece that represents certain important aspects of the way that the piece is intended to be perceived and interpreted.

Matches such as the ones in Fig. 1 can be found using fast, exact-matching algorithms if the pitch names of the notes are encoded, but exact-matching algorithms cannot be used to find such matches if the pitches are represented using just MIDI note numbers. If a reliable pitch spelling algorithm existed, it could be used to compute the pitch names of the notes in the tens of thousands of MIDI files of works that are freely available online, allowing these files to be searched more effectively by a music information retrieval (MIR) system.

1.2 Pitch Spelling in Common Practice Western Tonal Music

In the vast majority of cases, the correct pitch name for a note in a passage of tonal music can be determined by considering the rôles that the note plays in the harmonic, motivic and voice-leading structures of the passage. For example, when played in isolation in an equal-tempered tuning system, the first soprano note in Fig. 2(a) would sound the same as the first soprano note in Fig. 2(b).

However, in Fig. 2(a), this note is spelt as a G♯ because it functions as a leading note in A minor; whereas in Fig. 2(b), the first soprano note is spelt as an A♭ because it functions as a submediant in C minor. Similarly, the first alto note in Fig. 2(b) would sound the same as the first alto note in Fig. 2(c) in an equal-tempered tuning system. However, in Fig. 2(b) the first alto note is spelt as an F♮ because it functions in this context as a subdominant in C minor; whereas, in Fig. 2(c), the first alto note functions as a leading note in F♯ minor so it is spelt as an E♯.

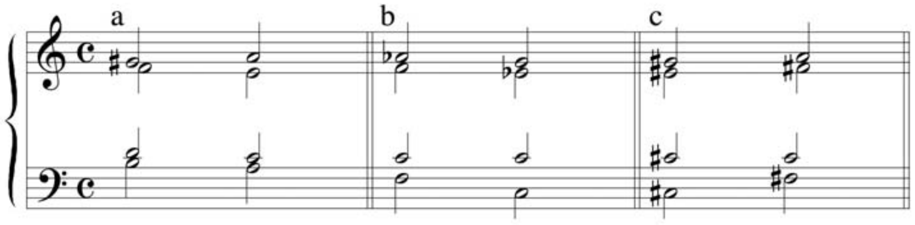


Fig. 2. Examples of enharmonic spelling (from [5, p. 8])

These examples illustrate that, in general, the pitch names assigned to notes in a passage of tonal music are not arbitrary. In most cases, they are carefully chosen so that the resulting score represents as well as possible certain important aspects of the way that the music is intended to be perceived and interpreted.

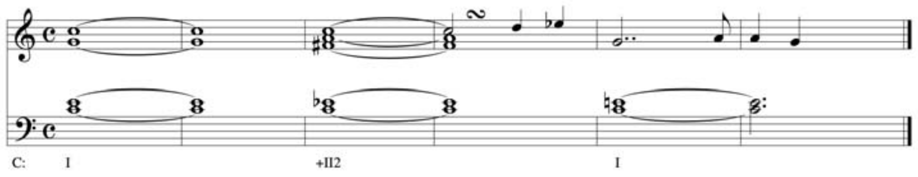


Fig. 3. Should the E♭s be spelt as D♯s? (From [5, p. 390])

Of course, there do exist cases where it is difficult to determine the correct pitch name of a note uniquely by considering the harmonic, motivic and voice-leading structures of its context. For example, as Piston observes [5, p. 390], the tenor E♭4 in the third and fourth bars of Fig. 3 should be spelt as a D♯4 if one perceives the harmonic progression here to be $^{+}\mathrm{II}^{2} - \mathrm{I}$ as proposed by Piston. But spelling the soprano E♭5 in the fourth bar as D♯5 would not represent the perceived structure of the melody correctly.

1.3 Using Accurate Encodings of Authoritative Editions as a 'Ground Truth'

Fortunately, however, cases such as the one in Fig. 3 where it is difficult to determine the correct pitch name of a note are relatively rare—particularly in Western tonal music of the so-called 'common practice' period (roughly the 18th and 19th centuries). In the vast majority of cases, those who study and perform Western tonal music agree about how a note should be spelt in a given tonal context. Correspondingly, the vast majority of notes in authoritative published editions of scores of common practice tonal works are generally agreed to be spelt correctly by those who understand Western staff notation.

Therefore a pitch spelling algorithm can be evaluated quantitatively by running it on tonal works and comparing the pitch names it predicts with those of the corresponding notes in authoritative published editions of scores of the works. In other words, such authoritative scores can provide us with a 'ground truth' that we can compare with the output of a pitch spelling algorithm.

However, this can only be done accurately and quickly if one has access to accurate encodings of these authoritative scores in the form of computer files that can be compared automatically with the pitch spelling algorithm's output. Currently there exist only a small number of publicly available collections of encodings of authoritative editions of musical scores (e.g., the MuseData collection (`http://www.musedata.org`) and the Mutopia Project (`http://www.mutopiaproject.org`)). However, if real progress is to be made in the development and testing of systems for music analysis, retrieval and transcription, it is necessary for much larger and more varied corpora of encoded scores to be made publicly available (or at least available for research purposes). Moreover, it is necessary for these corpora to be highly accurate. Unfortunately, building such corpora is currently an extremely time-consuming and error-prone process despite the existence of, for example, optical music recognition (OMR) systems (for an overview, see David Bainbridge's web page at `http://www.cs.waikato.ac.nz/~davidb/omr/`). There is therefore an urgent need for tools and techniques that will allow notated scores to be encoded more quickly and more accurately.

2 A Comparison of Three Pitch Spelling Algorithms

2.1 Introduction

In order to get a clearer idea of the 'state of the art' in the field, three pitch spelling algorithms were run on two test corpora and their performance was compared. The algorithms compared were those of Cambouropoulos [6, 7, 8, 9], Longuet-Higgins [10, 11, 12] and Temperley [13, 14]. In an initial pilot study [15], the corpus used was the first book of J. S. Bach's *Das Wohltemperirte Clavier*[1] (BWV 846–869), which contains 41544 notes. Then a larger-scale comparison

[1] This is Bach's own spelling of the work's title.

was carried out using a corpus containing 1729886 notes and consisting of 1655 movements from works by 9 baroque and classical composers (Corelli, Vivaldi, Telemann, Bach, Handel, B. Marcello, Haydn, Mozart and Beethoven). Both corpora were derived from the MuseData collection of encoded scores [16].

2.2 Longuet-Higgins's Algorithm

Pitch spelling is one of the tasks performed by Longuet-Higgins's `music.p` program [10, 11, 12]. The input to `music.p` must be in the form of a list of triples, $\langle p, t_{\mathrm{on}}, t_{\mathrm{off}} \rangle$, each triple giving the "keyboard position" p together with the onset time t_{on} and the offset time t_{off} in centiseconds of each note. The keyboard position p is just 48 less than the MIDI note number. So, for example, the keyboard position of middle C is 12. Longuet-Higgins intended the `music.p` program to be used only on monophonic melodies and explicitly warns against using it on "accompanied melodies" or what he calls "covertly polyphonic" melodies (i.e., compound melodies) [11, p. 114].

It is perhaps also worth pointing out that the pitch spelling algorithm implemented in `music.p` does *not* use Longuet-Higgins's well-known three-dimensional 'tonal space' model of tonality [11, p. 110–111]. However, Longuet-Higgins does actually describe this multi-dimensional model in the paper where the `music.p` program is published. This has probably led some readers to assume (incorrectly) that the program implements Longuet-Higgins's three-dimensional 'tonal space' model. In fact, the only pitch spaces used in the pitch spelling algorithm implemented in `music.p` are the circle of fifths and the 'line of fifths' (see Fig. 4).

	···	C♭	G♭	D♭	A♭	E♭	B♭	F	C	G	D	A	E	B	F♯	C♯	G♯	D♯	A♯	E♯	···
Sharpness	···	−7	−6	−5	−4	−3	−2	−1	0	1	2	3	4	5	6	7	8	9	10	11	···

Fig. 4. The line of fifths

The algorithm computes a value of "sharpness" for each note in the input [11, p. 111]. The sharpness of a note is a number indicating the position of the pitch name of the note on the line of fifths (see Fig. 4) [14, p. 117]. It is therefore essentially the same as Temperley's concept of *tonal pitch class* [14, p. 118] and Regener's concept of *quint* [17, p. 33].

Longuet-Higgins's algorithm tries to spell the notes so that they are as close as possible to the local tonic on the line of fifths [11, pp. 112–113]. The algorithm assumes at the beginning of the music that the first note is either the tonic or the dominant of the opening key and chooses between these two possibilities on the basis of the interval between the first two notes [11, p. 114]. The algorithm also disallows consecutive chromatic intervals [11, p. 113] and incorporates a special rule for dealing with ascending semitones [11, p. 114].

2.3 Cambouropoulos's Algorithm

The input to Cambouropoulos's method [6, 7, 8, 9] is again a sequence of MIDI note numbers in the order in which they occur in the music. The algorithm uses a "shifting overlapping windowing technique" [9, p. 420], as illustrated in Fig. 5, in which each window contains a certain number (typically 9 or 12) of contiguous elements in the input sequence.

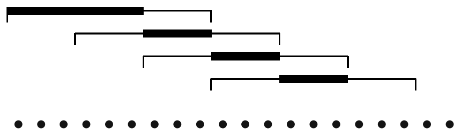


Fig. 5. Cambouropoulos's own caption to this figure reads: "Shifting overlapping window technique. For each window only the middle section of spelled pitches (bold line) is retained. Dots represent the pitches of the input sequence." (Reproduced (with minor corrections) from [9, p. 420, Fig. 6].)

On each step, the window position advances by a third of the window size, as shown in Fig. 5. Windowing improves the running time of the algorithm (from exponential to linear in the size of the input) and overlapping the windows avoids certain types of errors at window boundaries.

Cambouropoulos allows 'white note' pitch classes (i.e., 0, 2, 4, 5, 7, 9 and 11) to be spelt in three ways. For example, pitch class 0 can be spelt as B♯, C♮ or D♭♭. 'Black note' pitch classes can be spelt in two ways. For example, pitch class 6 can be spelt as F♯ or G♭ (see, for example, [6, p. 242]). Given these restricted pitch name possibilities for each note, Cambouropoulos's method computes all the spellings for each window that do not contain both double-sharps and double-flats. For example, for a 9-note window, 128 different spellings will have to be evaluated.

A penalty score is then computed for each of these possible window spellings. The penalty score for a given window spelling is found by computing a penalty value for each pitch interval in the window and summing these interval penalty values. A given interval in a particular window spelling is penalised more heavily the less frequently it occurs in the major and minor scales, a principle that Cambouropoulos calls *interval optimization* [9, p. 421]. An interval is also penalised if either of the pitch names forming the interval is a double-sharp or a double-flat, a principle that Cambouropoulos calls *notational parsimony* [9, p. 421]. For each window, the algorithm chooses the spelling that has the lowest penalty score and retains the pitch names for the middle third of this best window spelling.

2.4 Temperley's Algorithm

Temperley's pitch spelling algorithm [13, 14] is implemented in his `harmony` program which forms one component of his and Sleator's *Melisma* system. The input to the `harmony` program must be in the form of a “note-list” [14, pp. 9–12] giving the MIDI note number of each note together with its onset time and duration in milliseconds. This note list must be accompanied by a representation of the metrical structure of the passage in the form of a “beat-list” of the type generated by Temperley and Sleator’s `meter` program.

Temperley’s pitch spelling algorithm searches for the spelling that best satisfies three “preference rules” [14, pp. 115–136]. The first of these rules stipulates that the algorithm should “prefer to label nearby events so that they are close together on the line of fifths” [14, p. 125]. This rule bears some resemblance to the basic principle underlying Longuet-Higgins’s algorithm (see above). The second rule expresses the principle that if two tones are separated by a semitone and the first tone is distant from the key centre, then the interval between them should preferably be spelt as a diatonic semitone rather than a chromatic one [14, p. 129]. This rule is also very similar to one of the rules used in Longuet-Higgins’s algorithm. The third preference rule steers the algorithm towards spelling the notes so that what Temperley calls a “good harmonic representation” results [14, p. 131].

Note, however, that Temperley’s algorithm requires more information in its input than the other algorithms. In particular, it needs to know the duration of each note and the tempo at each point in the passage. It also needs to perform a full analysis of the metrical and harmonic structure of the passage in order to generate a high quality result. Also, it cannot deal with cases where two or more notes with the same pitch start at the same time.

2.5 Results of Running Algorithms on the First Book of J. S. Bach’s *Das Wohltemperirte Clavier*

Table 1 shows the results obtained in a pilot study in which the three algorithms described above were run on the first book of J. S. Bach’s *Das Wohltemperirte Clavier* [15]. As can be seen, Temperley’s algorithm performed best, followed by Longuet-Higgins’s algorithm and then Cambouropoulos’s.

Table 1. Results obtained when the three algorithms were run on the first book of J. S. Bach’s *Das Wohltemperirte Clavier*

Algorithm	*% notes correct*	*Number of errors*
Cambouropoulos	93.74	2599
Longuet-Higgins	99.36	265
Temperley	99.71	122

Total number of notes in corpus = 41544.

As mentioned above, Longuet-Higgins’s algorithm was not designed to be used on polyphonic music. Therefore, before running this algorithm on the corpus, each piece was first represented as a sequence of MIDI note numbers in the order in which they occur in the music. Within each set of note numbers corresponding to a set of notes beginning simultaneously, the elements were sorted into increasing order by note number.

3 The *ps13* Algorithm

3.1 Description of the Algorithm

Having carried out this pilot study and gained a better idea of the 'state of the art' in the field, an attempt was made to construct a new algorithm that improved on Temperley's. Around 30 different algorithms were developed and tested and the one that performed best on the first book of Bach's *Das Wohltemperirte Clavier* will now be described. This algorithm is called *ps13*.[2]

The input to *ps13* is, again, a sequence of MIDI note numbers in which the ordering corresponds to the order in which the notes occur in the music, any set of note numbers corresponding to notes that begin simultaneously being sorted into increasing order by note number. This sequence of MIDI note numbers is then converted into a sequence of *pitch classes* (if the MIDI note number of a note is n, then the pitch class of the note is $n \bmod 12$). At the highest level of description, *ps13* can be broken down into two stages, which I shall call *Stage 1* and *Stage 2*. In the following description, I denote by C the ordered set of pitch classes given to the algorithm as input. $C[i]$ denotes the $(i+1)$th element of C (e.g., $C[0]$ is the first element in C) and $|C|$ denotes the length of the sequence C.

Stage 1 involves carrying out the following steps:

Step 1 For each $0 \leq i < |C|$ and each pitch class $0 \leq p \leq 11$, compute a value $CNT(i,p)$ giving the number of times that p occurs within a context surrounding $C[i]$ that includes $C[i]$, K_{pre} notes immediately preceding $C[i]$ and $K_{\text{post}} - 1$ notes immediately following $C[i]$. K_{pre} is called the *precontext* and K_{post} is called the *postcontext*.

Step 2 For each $0 \leq i < |C|$ and each pitch class $0 \leq p \leq 11$, compute the number of diatonic steps $(\bmod 7)$, $D(i,p)$, that there would be from the tonic to the pitch name of $C[i]$ if p were the pitch class of the tonic at the point where $C[i]$ occurs. (Assume that the notes are spelt as they are in a harmonic chromatic scale whose tonic has pitch class p [21, p. 78]).

Step 3 For each $0 \leq i < |C|$ and each pitch class $0 \leq p \leq 11$, compute the value

$$D'(i,p) = (D(i,p) - D(0,p)) \bmod 7.$$

$D'(i,p)$ gives the number of diatonic steps $(\bmod 7)$ from the pitch name of the first note (i.e., the note corresponding to $C[0]$) to the pitch name of $C[i]$ if the tonic pitch class is p.

[2] Patent pending [18, 19, 20]. Please contact the author at `dave@titanmusic.com` if you wish to use the algorithm. Permission will usually be granted for free use of the algorithm for non-commercial purposes.

Step 4 For each $0 \leq i < |C|$ and each diatonic interval $0 \leq d \leq 6$, compute the set of tonic pitch classes,

$$X(i, d) = \{p \mid D'(i, p) = d\}.$$

$X(i, d)$ contains the tonic pitch classes that would lead to the diatonic interval from the first note to $C[i]$ being d.

Step 5 For each $0 \leq i < |C|$ and each diatonic interval $0 \leq d \leq 6$, compute the sum, $N(i, d)$, of the values of $CNT(i, p)$ for all the tonic pitch classes $p \in X(i, d)$. That is,

$$N(i, d) = \sum_{p \in X(i,d)} CNT(i, p).$$

Step 6 For each $0 \leq i < |C|$, compute $d_{\max}(i)$, the value of d for which $N(i, d)$ is a maximum.

Step 7 Assign a letter name to the first note, $C[0]$. This can be done, for example, by using the table below

$C[0]$	0	1	2	3	4	5	6	7	8	9	10	11
Letter name of $C[0]$	C	C	D	E	E	F	F	G	A	A	B	B

Step 8 For each $0 \leq i < |C|$, make the letter name of the note corresponding to $C[i]$, $d_{\max}(i)$ steps above the letter name assigned to the note corresponding to $C[0]$.

The actual letter name assigned to the first note in Step 7 is not critical. It might make the difference, for example, between a piece being spelt in A♭ rather than G♯ but it would not change the pitch interval name of the interval between any pair of notes in the piece. Clearly, if a correctly notated piece is transposed up or down by a diminished second, the resulting piece will still be correctly notated—it will just be written in a different key. In general, of course, the key is chosen so as to minimise the number of sharps or flats in the key signature as this is supposed to make the score easier to read. For example, a piece notated in C major would generally be regarded as being easier to read than the same piece notated in B♯ major.

Stage 2 of the algorithm corrects those instances in the output of *Stage 1* where a neighbour note or passing note is erroneously predicted to have the same letter name as either the note preceding it or the note following it. That is, the second stage of the algorithm corrects errors like those shown in Fig. 6.

In the first step of *Stage 1*, the algorithm essentially counts how many times each pitch class occurs within some specified context surrounding a particular note. Krumhansl [22, pp. 66–75] showed that there is a high correlation between the frequency with which a pitch class occurs within a passage and its perceived tonal stability as measured experimentally [23]. This suggests that the value

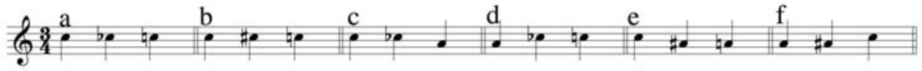


Fig. 6. Examples of the types of passing and neighbour note errors corrected in the second part of the *ps13* algorithm

CNT(i, p), calculated in the first step of *Stage 1*, gives an approximate measure of the perceived tonal stability of the pitch class p at the point in the music where $C[i]$ occurs. In *ps13* the value *CNT*(i, p) is used as a measure of the likelihood of p being perceived to be the tonic at the point where note $C[i]$ occurs. Note that, unlike Temperley's algorithm, *ps13* uses neither duration nor tempo and can deal with situations where two or more notes with the same pitch start at the same time.

3.2 Results of Running *ps13* on the First Book of J. S. Bach's *Das Wohltemperirte Clavier*

In the first step of *Stage 1*, *ps13* counts how many times each pitch class occurs within some specified context surrounding a particular note, defined by the values of K_{pre} and K_{post}. In order to explore the effect that varying the values of K_{pre} and K_{post} has on the performance of *ps13*, the algorithm was run 2500 times on the test corpus, each time using a different pair of values for K_{pre} and K_{post}, chosen so that both were between 1 and 50, inclusive. The image plot in Fig. 7 summarises the results obtained. It was found that *ps13* made fewer than 122 mistakes (i.e., performed better than Temperley's algorithm) on the test corpus for 2004 (80.160%) of the 2500 $\left\langle K_{\mathrm{pre}}, K_{\mathrm{post}}\right\rangle$ pairs tested.

ps13 performed best on the test corpus when K_{pre} was set to 33 and K_{post} was set to either 23 or 25. With these parameter values, *ps13* made only 81 errors on the test corpus—that is, it correctly predicted the pitch names of 99.81% of the notes in the test corpus. The mean number of errors made by *ps13* over all 2500 $\left\langle K_{\mathrm{pre}}, K_{\mathrm{post}}\right\rangle$ pairs was 109.082 (i.e., 99.74% of the notes were correctly spelt on average over all 2500 $\left\langle K_{\mathrm{pre}}, K_{\mathrm{post}}\right\rangle$ pairs). This average value was better than the result obtained by Temperley's algorithm for this test corpus (see areas in Fig. 7 corresponding to scores greater than 99.71%). The worst result was obtained when both K_{pre} and K_{post} were set to 1. In this case, *ps13* made 1117 errors (97.31% correct). However, provided K_{pre} was greater than about 13 and K_{post} was greater than about 16, *ps13* predicted the correct pitch name for over 99.71% of the notes in the test corpus.

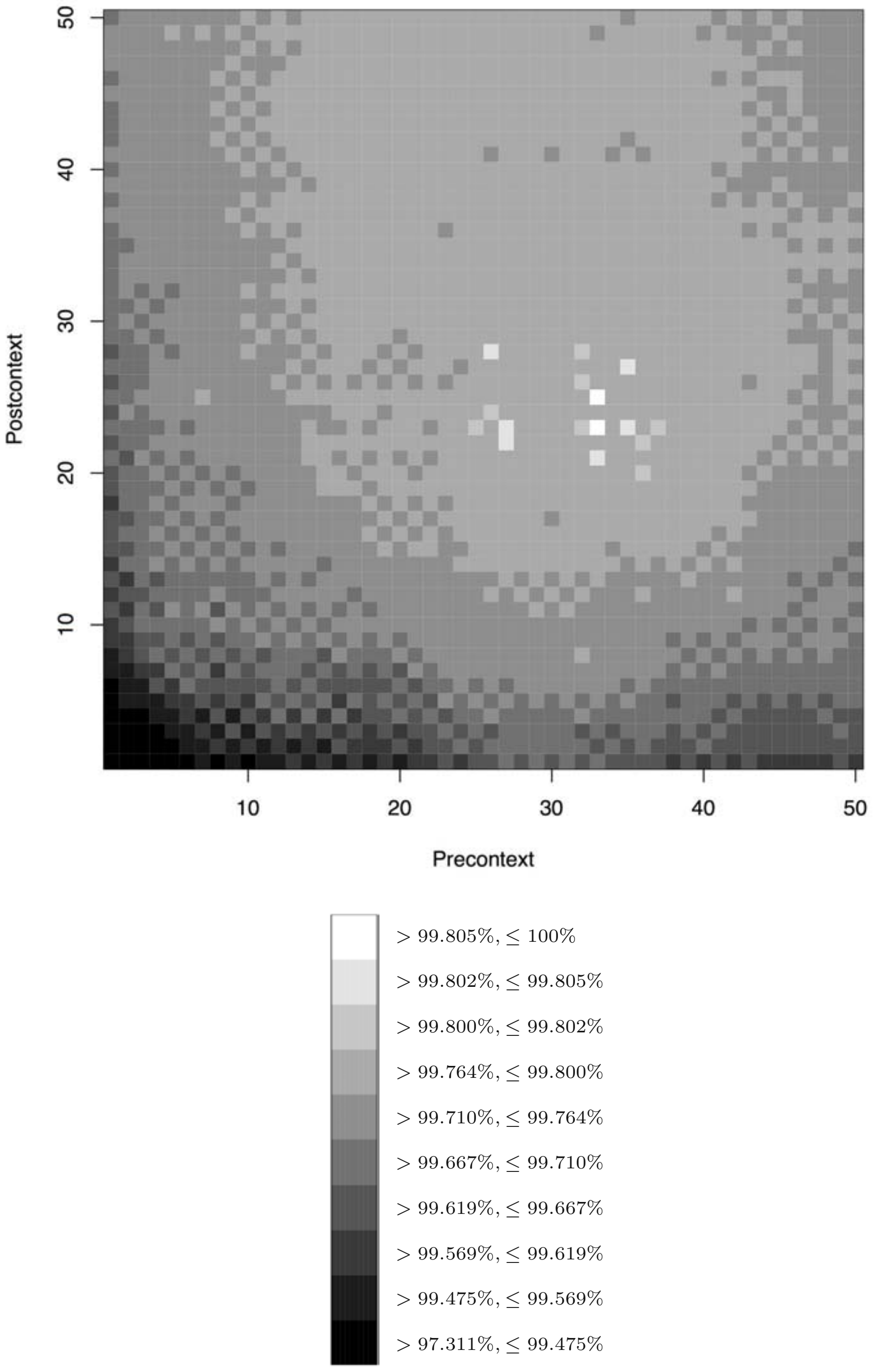


Fig. 7. Image plot showing percentage of notes spelt correctly by *ps13* for all values of K_{pre} and K_{post} between 1 and 50

4 Comparing the Algorithms on a Larger Corpus

4.1 Structure of the Larger Test Corpus

ps13 and the algorithms of Temperley, Cambouropoulos and Longuet-Higgins were then run on a much larger corpus containing 1729886 notes and consisting of 1655 movements from works by 9 baroque and classical composers (Corelli, Vivaldi, Telemann, Bach, Handel, B. Marcello, Haydn, Mozart and Beethoven). The values of K_{pre} and K_{post} for *ps13* were set to 33 and 23, respectively, these being the values that produced the best results when the algorithm was run on the smaller corpus in the pilot study. Table 2 shows the percentage and number of notes in this larger test corpus in works by each composer.

Table 2. Number and percentage of notes in large corpus in works by each composer

Composer	*Number of notes*	*% of notes*	*Cum. %*
Corelli	31390	1.81%	1.81%
Vivaldi	223678	12.93%	14.74%
Telemann	89542	5.18%	19.92%
Bach	627083	36.25%	56.17%
Handel	449793	26.00%	82.17%
Marcello	2962	0.17%	82.34%
Haydn	84682	4.90%	87.24%
Mozart	172097	9.95%	97.19%
Beethoven	48659	2.81%	100.00%
Total	1729886		

Table 2 reveals that more than 80% of the music in the corpus is baroque and the rest is classical. So the corpus is not very stylistically varied—it contains music written between about 1675 and 1825. Table 2 also shows that over 60% of the corpus consists of works by Bach and Handel. Indeed, the corpus is very unevenly distributed between the nine composers.

Unfortunately, a more varied and balanced corpus of a comparable size was not available at the time this experiment was carried out.

4.2 Comparison of Algorithms with Respect to Number of Notes Spelt Correctly

Table 3 shows the number of notes spelt incorrectly by each algorithm for each composer in the corpus. The bottom row in this table gives the total number of notes in the corpus spelt incorrectly by each algorithm.

Table 4 shows the percentage of notes spelt correctly by each algorithm for each composer. The bottom row of this table shows the total percentage of notes in the corpus spelt correctly by each algorithm. As shown in this table, overall, the percentage of notes spelt correctly is largest for the *ps13* algorithm.

Table 3. The number of notes spelt incorrectly by each algorithm for each composer

	Cam	LH	ps13	Tem
Corelli	148	120	30	6
Vivaldi	1816	5518	1497	4900
Telemann	604	665	481	111
Bach	13347	13022	3450	1166
Handel	1929	3342	2339	1309
Marcello	1	4	14	5
Haydn	1497	6174	823	6391
Mozart	2300	10147	2250	19832
Beethoven	600	1703	727	6525
Complete test corpus	22242	40695	11611	40245

Table 4. Percentage of notes spelt correctly by each algorithm for each composer

	Cam	LH	ps13	Tem
Corelli	99.53%	99.62%	99.90%	99.98%
Vivaldi	99.19%	97.53%	99.33%	97.81%
Telemann	99.33%	99.26%	99.46%	99.88%
Bach	97.87%	97.92%	99.45%	99.81%
Handel	99.57%	99.26%	99.48%	99.71%
Marcello	99.97%	99.86%	99.53%	99.83%
Haydn	98.23%	92.71%	99.03%	92.45%
Mozart	98.66%	94.10%	98.69%	88.48%
Beethoven	98.77%	96.50%	98.51%	86.59%
Complete test corpus	98.71%	97.65%	99.33%	97.67%

McNemar's test [24, 25] was then used to determine whether the differences between the scores achieved by the algorithms were statistically significant.[3] McNemar's test is essentially a chi-squared test for related samples. Let's denote by $S_{A,T}$ and $S_{B,T}$ the percentages of notes spelt correctly by algorithms A and B, respectively, when they are run on a test corpus T. Let us further assume that $S_{A,T} > S_{B,T}$. McNemar's test can be used to estimate the probability (*p-value*) of getting results at least as different as $S_{A,T}$ and $S_{B,T}$ if there would actually be no difference between the percentage of notes spelt correctly by A and B if they were run on the population from which T was taken. The first step in computing this p-value is to compute the values n_{ic}, the number of notes incorrectly spelt by A but correctly spelt by B; and n_{ci}, the number of notes correctly spelt by A but incorrectly spelt by B. If A and B would actually perform equally well on the whole population, then we expect n_{ci} to be equal to n_{ic}. McNemar's test assumes that the expected values of n_{ci} and n_{ic} are both equal to $(n_{ci} + n_{ic})/2$ and compares the observed values of n_{ci} and n_{ic} with this expected value. An estimate of χ^2 is then obtained using the following formula

$$\chi^2 = \frac{(|n_{ic} - n_{ci}| - 1)^2}{n_{ic} + n_{ci}}.$$

The results of this analysis are shown in Table 5. In this table, each value in a column headed 'p-value' gives the statistical significance (expressed as a proba-

[3] I am grateful to Marcus Pearce for suggesting the use of McNemar's test.

bility computed using McNemar's test) of the difference in performance between the algorithms to the left and right of the value in the table. For example, the value in the seventh column and first row of Table 5 indicates that if the algorithms of Longuet-Higgins and Cambouropoulos would actually make an equal number of note errors when run over the whole population of works represented by the test corpus, then the probability of getting a difference in scores at least as great as that observed in this study would be 0.0765. The bottom row of this table shows that, overall, *ps13* spelt very significantly more notes correctly than Cambouropoulos's algorithm ($p < 0.0001$), which in turn spelt very significantly more notes correctly than Temperley's algorithm ($p < 0.0001$). However, the percentage of notes spelt correctly by Temperley's algorithm (97.67%) and Longuet-Higgins's algorithm (97.65%) did not differ significantly ($p = 0.0954$). (A difference is considered significant if $p \leq 0.05$.)

Table 5. Statistical significance of differences between note error rates of algorithms for each composer and for complete corpus, measured using McNemar's test

Composer	Best	p-value	2nd best	p-value	3rd best	p-value	Worst
Corelli	Tem	< 0.0001	ps13	< 0.0001	LH	0.0765	Cam
Vivaldi	ps13	< 0.0001	Cam	< 0.0001	Tem	< 0.0001	LH
Telemann	Tem	< 0.0001	ps13	< 0.0001	Cam	0.0736	LH
Bach	Tem	< 0.0001	ps13	< 0.0001	LH	0.0352	Cam
Handel	Tem	< 0.0001	Cam	< 0.0001	ps13	< 0.0001	LH
Marcello	Cam	0.1797	LH	0.7388	Tem	0.0389	ps13
Haydn	ps13	< 0.0001	Cam	< 0.0001	LH	0.0348	Tem
Mozart	ps13	0.3665	Cam	< 0.0001	LH	< 0.0001	Tem
Beethoven	Cam	< 0.0001	ps13	< 0.0001	LH	< 0.0001	Tem
Complete test corpus	ps13	< 0.0001	Cam	< 0.0001	Tem	0.0954	LH

The graph in Fig. 8 shows the percentage of notes spelt correctly by each algorithm for each composer, the composers being placed along the horizontal axis in increasing chronological order of birth. This graph suggests that the algorithms of Temperley and Longuet-Higgins perform significantly worse on the classical composers (Haydn, Mozart and Beethoven) than they do on the baroque composers. The graph also suggests that *ps13* performs more consistently across the different composers and styles than the other algorithms.

4.3 Comparison of Algorithms with Respect to Number of Intervals Spelt Correctly

When the lists of errors generated by the algorithms were examined, it was observed that, in some cases, many errors were the result of large segments of the music simply being transposed up or down by a diminished second. In other words, a single incorrect interval between two consecutive notes in the input representation resulted in a whole segment of notes following the incorrect interval being spelt incorrectly. The algorithms were therefore also compared

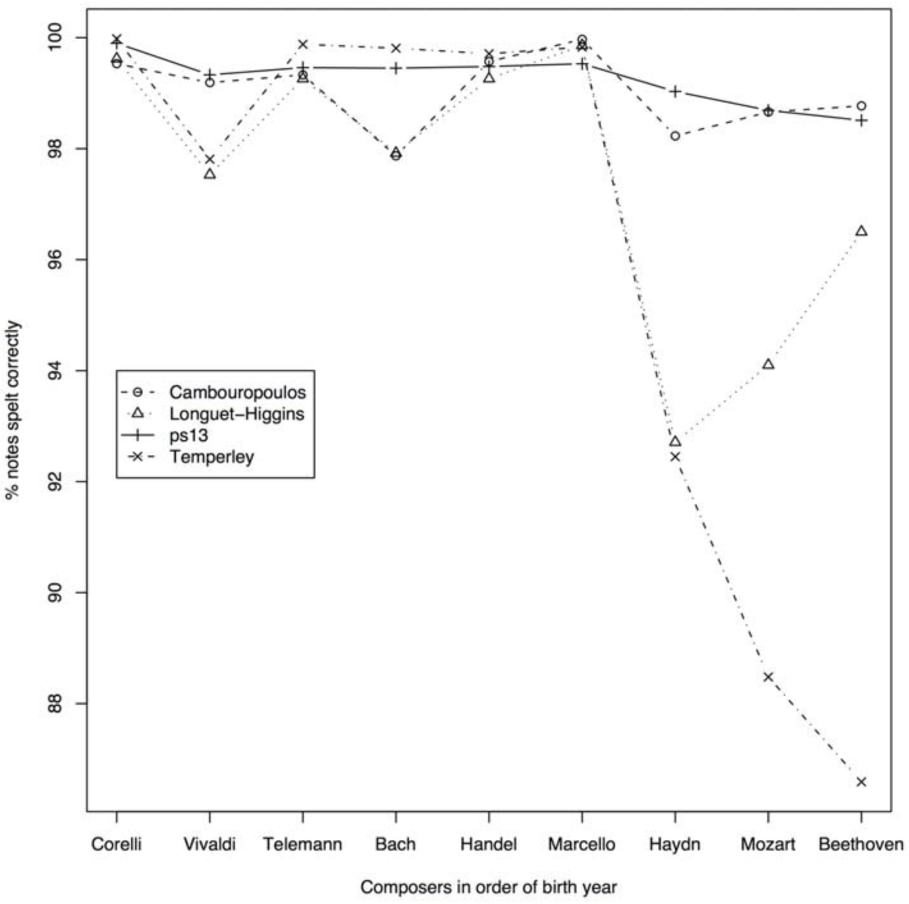


Fig. 8. Graph showing the percentage of notes spelt correctly by each algorithm for each composer, with the composers arranged along the horizontal axis in increasing chronological order of birth

with respect to the number of *intervals* spelt correctly between consecutive notes in the input representations. Table 6 shows the number of intervals between consecutive notes in works by each composer in the corpus.

The graph in Fig. 9 shows the percentage of intervals between consecutive notes spelt correctly by each algorithm for each composer, the composers being placed along the horizontal axis in increasing chronological order of birth. From this graph it is clear that the algorithms of Longuet-Higgins and Temperley were better at spelling *intervals* between consecutive notes correctly than they were at spelling *notes* correctly. In particular, this graph shows that most of the note spelling errors made by Longuet-Higgins and Temperley's algorithms on the music of Haydn, Mozart and Beethoven were due to whole segments being spelt a diminished second away from the correct spelling.

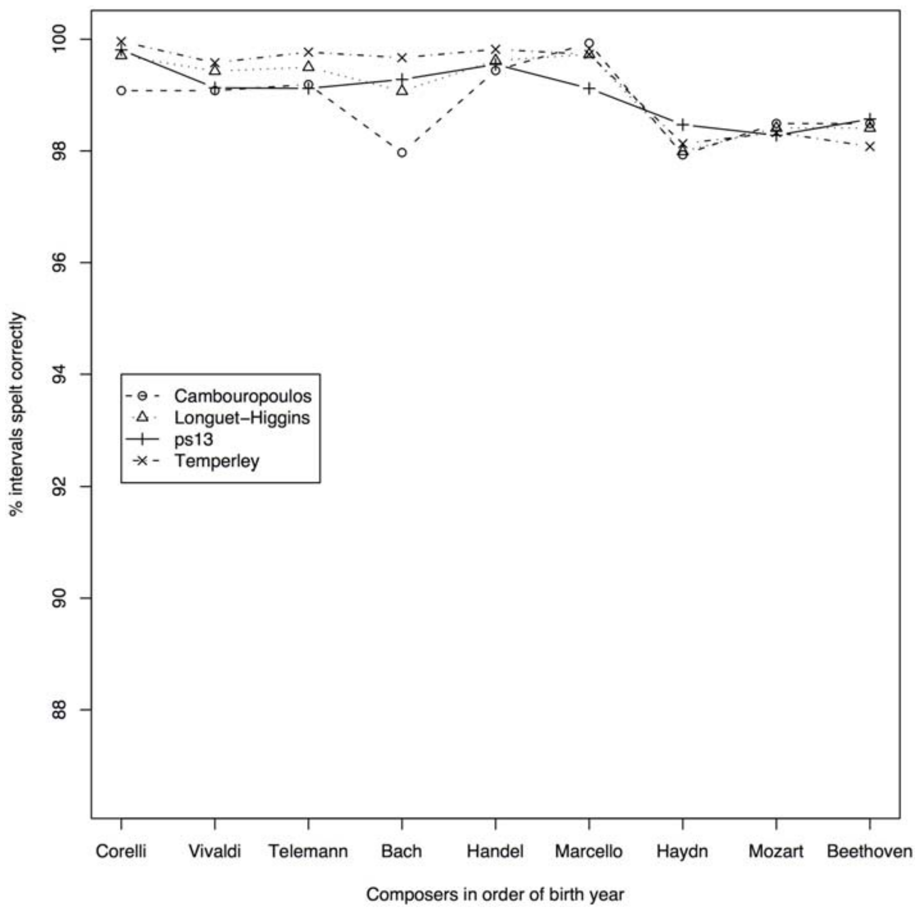


Fig. 9. Graph showing the percentage of intervals between consecutive notes spelt correctly by each algorithm for each composer, with the composers arranged along the horizontal axis in increasing chronological order of birth

Again, the statistical significance of the differences between the performances of the algorithms were analysed using McNemar's test and the results are shown in Table 9. The bottom line of this table reveals that, with respect to the number of intervals spelt correctly, Temperley's algorithm performs significantly better on this test corpus than the other three algorithms.

Nonetheless, the graph in Fig. 9 suggests that, even in terms of the number of intervals spelt correctly, all the algorithms perform worse on the classical music than on the baroque music.

Table 7 shows the percentage of intervals spelt correctly by each algorithm for each composer and Table 8 shows the number of intervals spelt incorrectly by each algorithm for each composer.

Table 6. The number of intervals between consecutive notes in the input representations in works by each composer in the test corpus

Corelli	Vivaldi	Telemann	Bach	Handel	Marcello	Haydn	Mozart	Beethoven	Complete
31302	223516	89385	626439	449292	2959	84648	172041	48649	1728231

Table 7. Percentage of intervals between consecutive notes in the input representations spelt correctly by each algorithm for each composer

	Cam	LH	ps13	Tem
Corelli	99.08%	99.71%	99.81%	99.96%
Vivaldi	99.08%	99.43%	99.13%	99.58%
Telemann	99.19%	99.50%	99.12%	99.77%
Bach	97.97%	99.07%	99.28%	99.67%
Handel	99.44%	99.62%	99.55%	99.82%
Marcello	99.93%	99.73%	99.12%	99.73%
Haydn	97.93%	97.99%	98.47%	98.13%
Mozart	98.49%	98.41%	98.28%	98.33%
Beethoven	98.49%	98.41%	98.57%	98.08%
Complete test corpus	98.65%	99.16%	99.17%	99.45%

Table 8. Number of intervals between consecutive notes in the input representations spelt incorrectly by each algorithm for each composer

	Cam	LH	ps13	Tem
Corelli	288	92	60	12
Vivaldi	2064	1264	1937	941
Telemann	727	450	788	204
Bach	12697	5852	4507	2074
Handel	2518	1703	2030	822
Marcello	2	8	26	8
Haydn	1750	1700	1298	1579
Mozart	2596	2734	2955	2874
Beethoven	736	772	695	933
Complete test corpus	23378	14575	14296	9447

Table 9. Statistical significance of differences between interval error rates of algorithms for each composer and for complete corpus, measured using McNemar's test

Composer	Best	p-value	2nd best	p-value	3rd best	p-value	Worst
Corelli	Tem	< 0.0001	ps13	0.0068	LH	< 0.0001	Cam
Vivaldi	Tem	< 0.0001	LH	< 0.0001	ps13	0.1077	Cam
Telemann	Tem	< 0.0001	LH	< 0.0001	Cam	0.0029	ps13
Bach	Tem	< 0.0001	ps13	< 0.0001	LH	< 0.0001	Cam
Handel	Tem	< 0.0001	LH	< 0.0001	ps13	< 0.0001	Cam
Marcello	Cam	0.0577, 0.0577	LH,Tem	0.001, 0.002	ps13		
Haydn	ps13	< 0.0001	Tem	0.0028	LH	0.2416	Cam
Mozart	Cam	0.0584	LH	0.0143	Tem	0.2459	ps13
Beethoven	ps13	0.041	Cam	0.0252	LH	< 0.0001	Tem
Complete test corpus	Tem	< 0.0001	ps13	0.0637	LH	< 0.0001	Cam

5 Conclusions and Further Work

This paper presents the results of a study in which four pitch spelling algorithms were compared on a large corpus of tonal music. The algorithms compared were those of Cambouropoulos, Temperley and Longuet-Higgins, together with the author's own *ps13* algorithm. When the algorithms were evaluated in terms of the number of notes in this corpus that they spelt correctly, it was found that the author's *ps13* algorithm performed significantly better than the other algorithms, correctly spelling 99.33% of the notes in the corpus correctly. While all four algorithms performed well on the baroque music in the test corpus, it was found that the algorithms that were directly based on the circle of fifths (i.e., those of Temperley and Longuet-Higgins) performed less well than the other algorithms on the classical music in the corpus. However, when the algorithms were evaluated in terms of the number of *intervals* between consecutive notes spelt correctly, it was found that all the algorithms performed very well across all styles, with Temperley's algorithm performing best, correctly spelling 99.45% of the intervals in the corpus correctly.

It would be interesting to extend this study by doing further comparisons between the algorithms described here and other algorithms such as the real-time algorithms of Chew and Chen [26] and also the algorithms implemented in commercial music notation programs such as Sibelius (`http://www.sibelius.com`) and Finale (`http://www.finalemusic.com`). These algorithms should also be tested on other corpora containing works in a wider variety of tonal styles including, for example, romantic, impressionist, rock and jazz music.

Krumhansl [22, p. 79] claims that "once a key (or key region) has been determined, the correct spellings of the tones will be able to be determined in most cases". *ps13* and the algorithms of Temperley and Longuet-Higgins all perform something at least very much like key finding as part of the pitch-spelling process. However, these algorithms give different results when run on the same test corpora. This demonstrates that there are various plausible ways of using the key-structure of a passage to determine pitch names and it is not at all obvious which of these methods will give the best results. Krumhansl's claim needs to be tested by building complete pitch spelling algorithms based on various key-finding algorithms and comparing the performance of these algorithms with others such as those described in this paper.

Finally, the errors made by the algorithms in the experiments described above need to be analysed in more depth in order to determine if any further improvements can be made.

Acknowledgements

While carrying out the research reported in this paper, the author was funded by the EPSRC (grant numbers GR/N08049/01 and GR/S17253/01). The author would like to thank Geraint Wiggins, Tim Crawford, Marcus Pearce and the other members of the ISMS group at City University, London (`http://ccc.soi.city.ac.uk/isms/`) for their kind and helpful suggestions and support.

References

[1] Davy, M., Godsill, S.J.: Bayesian harmonic models for musical signal analysis (with discussion). In Bernardo, J.M., Berger, J.O., Dawid, A.P., Smith, A.F.M., eds.: Bayesian Statistics. Volume VII. Oxford University Press (2003) Draft available online at `http://www-sigproc.eng.cam.ac.uk/~sjg/papers/02/harmonicfinal2.ps`.

[2] Walmsley, P.J.: Signal Separation of Musical Instruments. PhD thesis, Signal Processing Group, Department of Engineering, University of Cambridge (2000)

[3] Plumbley, M., Abdallah, S., Bello, J., Davies, M.E., Monti, G., Sandler, M.: Automatic music transcription and audio source separation. Cybernetics and Systems **33** (2002) 603–627

[4] Meredith, D., Lemström, K., Wiggins, G.A.: Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. Journal of New Music Research **31** (2002) 321–345 Draft available online at `http://www.titanmusic.com/papers/public/siajnmr_submit_2.pdf`.

[5] Piston, W.: Harmony. Victor Gollancz Ltd., London (1978) Revised and expanded by Mark DeVoto.

[6] Cambouropoulos, E.: A general pitch interval representation: Theory and applications. Journal of New Music Research **25** (1996) 231–251

[7] Cambouropoulos, E.: Towards a General Computational Theory of Musical Structure. PhD thesis, University of Edinburgh (1998) Available online at `http://users.auth.gr/~emilios/englishpage/phd.html`.

[8] Cambouropoulos, E.: Automatic pitch spelling: From numbers to sharps and flats. In: VIII Brazilian Symposium on Computer Music (SBC&M 2001), Fortaleza, Brazil (2001) Available online at `ftp://ftp.ai.univie.ac.at/papers/oefai-tr-2001-12.pdf`.

[9] Cambouropoulos, E.: Pitch spelling: A computational model. Music Perception **20** (2003) 411–429

[10] Longuet-Higgins, H.C.: The perception of melodies. Nature **263** (1976) 646–653

[11] Longuet-Higgins, H.C.: The perception of melodies. In Longuet-Higgins, H.C., ed.: Mental Processes: Studies in Cognitive Science. British Psychological Society/MIT Press, London, England and Cambridge, Mass. (1987) 105–129

[12] Longuet-Higgins, H.C.: The perception of melodies. In Schwanauer, S.M., Levitt, D.A., eds.: Machine Models of Music. M.I.T. Press, Cambridge, Mass. (1993) 471–495

[13] Temperley, D.: An algorithm for harmonic analysis. Music Perception **15** (1997) 31–68

[14] Temperley, D.: The Cognition of Basic Musical Structures. MIT Press, Cambridge, MA (2001)

[15] Meredith, D.: Pitch spelling algorithms. In Kopiez, R., Lehmann, A.C., Wolther, I., Wolf, C., eds.: Proceedings of the Fifth Triennial ESCOM Conference (ESCOM5) (8-13 September 2003), Hanover University of Music and Drama, Hanover, Germany. (2003) pp. 204–207 Draft available online at `http://www.titanmusic.com/papers/public/ps13-escom-paper.pdf`.

[16] Hewlett, W.B.: *MuseData*: Multipurpose representation. In Selfridge-Field, E., ed.: Beyond MIDI: The Handbook of Musical Codes. MIT Press, Cambridge, MA. (1997) 402–447

[17] Regener, E.: Pitch Notation and Equal Temperament: A Formal Study. University of California Press, Berkeley, CA. (1973)

[18] Meredith, D.: Method of computing the pitch names of notes in MIDI-like music representations (2003) Patent filing submitted to UK Patent Office on 11 April 2003. Application number 0308456.3. Draft available online at `http://www.titanmusic.com/papers/public/ps13-patent-1.pdf`.

[19] Meredith, D.: Method of computing the pitch names of notes in MIDI-like music representations (2004) Patent filing submitted to UK Patent Office on 18 March 2004. Application number 0406166.9. Priority date 11 April 2003. Draft available online at `http://www.titanmusic.com/papers/public/ps13-patent-2.pdf`.

[20] Meredith, D.: Method of computing the pitch names of notes in MIDI-like music representations (2004) Patent filing submitted to US Patent Office on 12 April 2004. Application number 10/821962. Priority date 11 April 2003. Draft available online at `http://www.titanmusic.com/papers/public/us-ps13-patent.pdf`.

[21] Associated Board of the Royal Schools of Music: Rudiments and Theory of Music. Associated Board of the Royal Schools of Music, 14 Bedford Square, London, WC1B 3JG (1958)

[22] Krumhansl, C.L.: Cognitive Foundations of Musical Pitch. Volume 17 of Oxford Psychology Series. Oxford University Press, New York and Oxford (1990)

[23] Krumhansl, C.L., Kessler, E.J.: Tracing the dynamic changes in perceived tonal organisation in a spatial representation of musical keys. Psychological Review **89** (1982) 334–368

[24] McNemar, Q.: Psychological Statistics. 4th edn. John Wiley and Sons, New York (1969)

[25] Dietterich, T.G.: Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation **10** (1998) 1895–1924

[26] Chew, E., Chen, Y.C.: Determining context-defining windows: Pitch spelling using the spiral array. In: Fourth International Conference on Music Information Retrieval (ISMIR 2003), Baltimore, MD. (2003) Available online at `http://www-rcf.usc.edu/~echew/papers/ISMIR4/ecyc-ismir4.pdf`.

Score-PCM Music Synchronization Based on Extracted Score Parameters

Vlora Arifi, Michael Clausen, Frank Kurth, and Meinard Müller

Universität Bonn, Institut für Informatik III
Römerstr. 164, D-53117 Bonn, Germany
{arifi, clausen, kurth, meinard}@cs.uni-bonn.de

Abstract. In this paper we present algorithms for the automatic time-synchronization of score-, MIDI- or PCM-data streams which represent the same polyphonic piano piece. In contrast to related approaches, we compute the actual alignment by using note parameters such as onset times and pitches. Working in a score-like domain has advantages in view of the efficiency and accuracy: due to the expressiveness of score-like parameters only a small number of such features is sufficient to solve the synchronization task. To obtain a score-like representation from the waveform-based PCM-data streams we use a preprocessing step to extract note parameters. In this we use the concept of novelty curves for onset detection and multirate filter banks in combination with note templates for pitch extraction. Also the data streams in MIDI- and score-format have to be suitably preprocessed. In particular, we suggest a data format which handles possible ambiguities such as trills or arpeggios by introducing the concept of fuzzy-notes. Further decisive ingredients of our approach are carefully designed cost functions in combination with an appropriate notion of alignment which is more flexible than the classical DTW concept. Our synchronization algorithms have been tested on a variety of classical polyphonic piano pieces recorded on MIDI- and standard acoustic pianos or taken from CD-recordings.

1 Introduction

Modern digital music libraries consist of large collections of documents containing music data of diverse characteristics and formats. For example, for one and the same piece of music, the library may contain the corresponding score in the Capella or Score format, as MIDI-files, and several interpretations by various musicians as CD recordings. Inhomogeneity and complexity of such music data make content-based browsing and retrieval in digital music libraries a difficult task with many yet unsolved problems. One important step towards a solution are synchronization algorithms which automatically align data streams of different data formats representing a similar kind of information. In particular, in the framework of audio by *synchronization* we mean a procedure which, for a given position in some representation of a given piece of music (e.g., given in score format), determines the corresponding position within some other representation (e.g., given in PCM-format).

Such synchronization algorithms have applications in many different scenarios: following score-based music retrieval, linking structures can be used to access a suitable audio CD accurately to listen to the desired part of the interpretation. A further future application is the automatic annotation of a piece of music in different data formats to support content-based retrieval and browsing. As another example, musicologists can use synchronizations for the investigation of agogic and tempo studies. Furthermore, temporal linking of score and audio data can be useful for a reading aid of scores.

In the following, we concentrate on three widely used formats for representing music data: the symbolic *score format* contains information on the notes such as musical onset time, pitch, duration, and further hints concerning the agogic and dynamics. The purely physical *PCM-format* encodes the waveform of an audio signal as used for CD-recordings. The *MIDI-format* may be thought of as a hybrid of the last two data formats containing content-based information on the notes as well as agogic and dynamic niceties of some specific interpretation. These different data formats lead to various synchronization problems as illustrated by Figure 1.

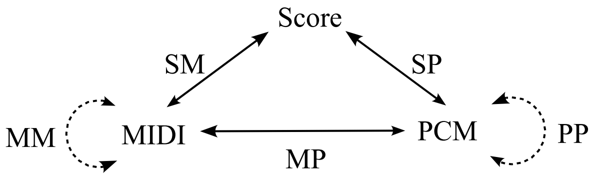


Fig. 1. Overview on various synchronization problems.

In this paper we concentrate on the case of Score-PCM (SP) synchronization where one data stream is given in a score-like format (in this case the note parameters are explicitly available) and the other data stream is given in the PCM-format. The other cases such as SM- or MP-synchronization are even easier or can be done in a similar fashion (see [1]).

As will be discussed in Section 2 there are various strategies to make the score-like data stream comparable to the PCM-data stream. In all approaches the data streams are divided up into sequences of frames which are aligned by techniques of dynamic time warping (DTW). Since the memory requirement as well as the running time of this approach is at least proportional to the product of the lengths of the two sequences to be aligned, efficiency becomes an important issue. Therefore, in contrast to the approaches described in [17,18], we suggest to use score-like features such as onset times and pitches for the alignment process. Due to the expressiveness of such note parameters only a small number of features is sufficient to solve the synchronization task. Furthermore, a very good temporal accuracy of the alignment can be achieved.

The major problem with this approach is that a PCM-data stream does not explicitly contain information on the notes. Therefore, in a preprocessing step, we first extract information such as note onsets and pitches from the PCM-recording in order to make it comparable in the score-like domain. However, the extraction of note information from the waveform of polyphonic music constitutes an extremely difficult problem which is solved only for a few special cases (see also Section 2). Especially, in the most general case of orchestral music the extraction problem (not to mention the transcription problem) is a largely open problem which seems to be unfeasible. In our research, we have concentrated on polyphonic piano music. In contrast to many other research projects we do not restrict ourselves to PCM-data generated by MIDI-pianos. Instead we allow PCM-data generated by any acoustic piano, e.g., music data from a piano CD. This in general extremely complex data leads to many erroneous extraction results which would not be acceptable when treated as a transcription into a score-like format of the original piece of music. However, the extracted data — even from complex piano pieces — is good enough to ensure success in view of the synchronization problem.

One major problem in Score-PCM synchronization results from the fact that the score is just a description of the piece of music which leaves a lot of room for various interpretations not only concerning the tempo and dynamics but also concerning the notes itself. To cope with ambiguities in the score, such as trills, arpeggios, or grace notes, we introduce the concept of fuzzy-notes which allows a set of alternative notes in the alignment process. Furthermore, concerning the PCM-based interpretation one often has to deal with wrong, additional, or missing notes. Using classical DTW-based approaches for the alignment, as e.g. described in [13,17,18], matches are forced by the local constraints imposed on the warping paths — even in regions where there are no equivalent events in the score- and PCM-data streams. From our experiments we conclude that it is preferable to avoid such kinds of matches rather than having "erroneous" matches. To this means, we introduce a more general notion of a so-called *Score-PCM match* which allows to skip notes and whole regions, in which the PCM-version differs considerably from the score.

The rest of this paper is organized as follows. In Section 2, we discuss recent approaches to Score-PCM synchronization and briefly review some literature related to the extraction and transcription problem. Then, in Section 3, we summarize our system for the extraction step of musically relevant parameters from PCM-data streams. The actual synchronization algorithms, based on a carefully designed cost function, are described in Section 4. Finally, Section 5 contains an examples, Section 6 summarizes some of our experiments, and Section 7 closes with concluding remarks and possible future research directions.

2 Background and Related Work

The synchronization problem in music and in particular the related problems of automatic score following and automatic music accompaniment has been a research field for many years. Dannenberg et. al. describe in [3] and [4] one of the first algorithms for automatic music accompaniment reducing the synchronization problem to an LCS (longest common subsequence) problem which is solved using dynamic programming. Raphael [14] has developed a system for automatic musical accompaniment of an oboe based on Hidden Markov Models. Desain et. al. describe in [5] a general sequential and tree-based score-performance matching algorithms. A similar problem addresses Large in [9] using dynamic programming to study music production errors. In all of those approaches the data streams involved in the synchronization problem either explicitly contain score-like note parameters or only consist of monophonic music so that the note parameters are comparatively clean and error free. In our scenario, however, we also allow PCM-data of complex polyphonic piano music with no such explicit and clean parameters. Further links may be found in the overview article [12]. In the following, we discuss two recent contributions by Turetsky et al. [18] and by Soulez et al. [17] which also address the problem of Score-PCM synchronization.

To make the data sequences comparable, Turetsky et al. [18] first convert the score-like version (given as MIDI data stream) into a PCM-data stream using a suitable synthesizer. Then, the two PCM-data streams are analyzed by means of a Short-Time-Fourier Transform (STFT) which in turn yields a sequence of suitable feature vectors. A pairwise comparison of those feature vectors with respect to a suitable local distance measure is used to compute a cost matrix. Based on this matrix, the best alignment is derived by means of dynamic time warping. As reported in [18], good synchronization results are achieved in the case of commercial pop music. In this genre, one generally has a relatively constant tempo and also a clear rhythm resulting in regular patterns in the spectral domain. Actually, Turetsky et al. convert the Score-PCM synchronization problem into a PCM-PCM synchronization problem — in the alignment process the note parameters given by the score-data stream are not used. On the one hand this makes their algorithm applicable for synchronizing arbitrary types of music. On the other hand, when dealing with e.g. classical music, where one can have strong local time deviations and only small spectral variations, algorithms relying solely on the comparison of the spectral information of the underlying PCM-data streams will lead to significant synchronization error rates.

In the approach of Soulez et al. [17] the score-data stream is used to generate a sequence of attack-, sustain- and silence models corresponding to note onsets, pitches and rests. As in [18], the PCM-data stream is converted into a sequence of spectral vectors using a STFT. Based on suitably defined local distance measures, which allow a comparison between the note models and the spectral vectors, a cost matrix is computed. Again dynamic time warping is applied to derive the alignment. In contrast to [18], Soulez et al. explicitly use the note parameters such as onset times and pitches in the synchronization process which results in a more robust algorithm w.r.t. local time deviations and small spectral variations.

Both approaches [17] and [18] have the following drawbacks due to the STFT used for the analysis of the PCM-data stream. Firstly, the STFT computes spectral coefficients which are *linearly* spread over the spectrum resulting in a bad resolution of low frequencies. Therefore, in the case of low notes one has to rely on the harmonics. This is problematic in polyphonic music where one often has the situation that harmonics and fundamental frequencies of different notes coincide. Secondly, in order to obtain a satisfying time resolution one has to work with a relatively large number of feature vectors on the PCM-side. (For example, even with a rough time resolution of 46 ms as suggested in [18] these are more than 20 feature vectors per second.) This leads to huge memory requirements as well as long running times in the DTW computation, which are both proportional to the product of the lengths of the two sequences to be aligned.

As mentioned in the last section, the extraction problem of score-like note parameters from waveform-based PCM-data itself constitutes an active research area with many yet unsolved problems. We only give a small choice of relevant literature where the reader finds further links. As an example, we mention the approach of Raphael [15] who uses Hidden Markov Models to transcribe polyphonic piano music. Klapuri et. al. [10] use moving-average techniques to extract note pitches in polyphonic musical signals. As is also mentioned by the authors, the extraction of such parameters in polyphonic music still leaves a lot of work to do. Foote [6] uses the concept of the so-called novelty score to automatically segment audio recordings. In Section 3 we use a similar technique to extract candidates of onset times. Bobrek et. al. [2] use filter bank techniques in combination with note templates for the transcription of polyphonic piano music. We have modified their approach to extract the pitches of the previously determined onset candidates. Finally, we want to mention the comprehensive book [11] by Mazzola who gives, among many related topics, a detailed account on local tempo variations resulting from expressiveness in performances.

3 Feature Extraction

In this section we summarize our system for extracting note parameters from the PCM-data stream. Using several established tools from audio signal processing, our main contributions are a refined template matching algorithm for polyphonic pitch extraction and a two-step algorithm for note onset detection.

Figure 2 shows the overall feature extraction algorithm. An input PCM-signal is transformed to a subband representation using a multirate filterbank. Simultaneously, a two-stage peak-picking algorithm detects probable note onset positions. According to those onset positions, the subband representation is split into time intervals. For each interval, we calculate an energy vector with components corresponding to the subbands: each component contains the total energy within the interval of the respective subband. Then, for each energy vector a pitch extraction based on a template matching algorithm is performed. The pitch extraction yields a set of notes for the corresponding time interval. The feature extraction algorithm outputs a note object for each note in each time interval, where a note object consists of an onset time and a pitch information.

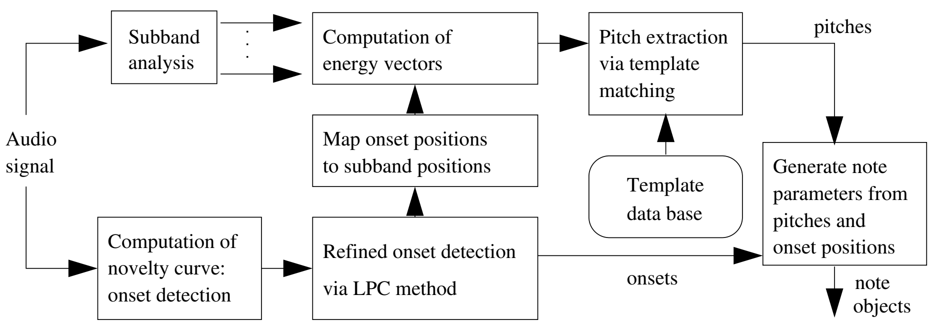

Subband analysis
Computation of energy vectors
Pitch extraction via template matching
pitches
Audio signal
Map onset positions to subband positions
Template data base
Generate note parameters from pitches and onset positions
Computation of novelty curve: onset detection
Refined onset detection via LPC method
onsets
note objects

Fig. 2. Diagram of the feature extraction algorithm.

We now briefly discuss the components of our algorithm. For onset detection, we first calculate a *novelty*[1] *curve* of the input signal. This step basically consists of a STFT with a step size of 23 ms, where for each step a novelty value is calculated by summing over the absolute changes of the last two steps' magnitude spectra. Peaks in the resulting novelty curve constitute candidate onset times. As the time-resolution of 23 ms is very rough, a postprocessing of all candidate onset-times is performed based on linear prediction. In linear predictive coding (LPC), the harmonic parts of a signal segment are summarized in a few prediction coefficients which may in turn be used to predict the short-time signal behavior. A method proposed in [7] compares ratios of prediction errors resulting from crossover predictions of neighboring signal segments to detect significant signal changes. We adopt this method to increase the time resolution of our candidate onset times to about 10 ms. For a detailed account we refer to [1].

The simultaneously applied subband filterbank transforms the input signal into $M = 224$ subband signals. The filterbank is realized by a tree structured cascade of orthogonal 2-band filterbanks. The tree structure of the filterbank — and hence the frequency ranges of the subbands — are chosen such that the fundamental frequency of at most one piano note (well-tempered tuning) falls into one subband. This guarantees that in the subsequent template matching algorithm templates can be uniquely assigned to energy vectors. For further details on the filter bank tree structure we refer to [2].

For pitch detection, template matching is performed w.r.t. a template data base (TDB) consisting of one template for each musical note. A template is an M-point vector representing the energy distribution of a certain note over the above subbands. The templates were recorded using a Yamaha GranTouch E-Piano. We furthermore evaluated templates generated by two acoustic pianos (Steinway and Schimmel). However, the E-Piano's templates turned out to be the most robust for our purposes.

[1] We adopted the term *novelty* from [6] even though our definition of *novelty curve* does only conceptually correspond to the one used by Foote.

Starting with an initial energy vector, our template matching algorithm roughly tries to select a template from the TDB which optimally fits the energy vector w.r.t. a certain criterion. If successful, a corresponding energy fraction is subtracted from the energy vector to yield a modified vector, which is used to recurse the procedure until the remaining residual energy falls under a specified lower bound. We used the following criterion for selecting a template which optimally fits an energy vector $E \in \mathbb{R}^M$: find the lowest subband index k such that $E(k) \geq (c/M) \sum_{i=1}^{M} |E(i)|$ (for a suitable prior constant c). If such a k exists, search the TDB for a template S_k with fundamental frequency in this subband. If E contains S_k to a significant extent, select S_k as a matching template. In [2] the authors, instead of using the lowest significant subband, choose the subband containing the highest energy component for selecting the template. However, in our experiments this criterion turned out to yield several octave interval errors in pitch detection, especially when applied to recordings from acoustic pianos.

To conclude this section we note that our algorithms require a careful choice of parameters (e.g., thresholds for template extraction, peak picking, or the minimum inter-onset interval). A detailed discussion is beyond the scope of this paper. For further details, we refer to [1].

4 Synchronization Algorithms

In this section we describe the actual synchronization algorithms. Due to space limitation, we only consider the case of a score- and a PCM-data stream (SP-synchronization). The other cases such as SM- or MP-synchronization are even easier or can be done in a similar fashion (see [1]).

First, we discuss how to preprocess the score data which is assumed to exist in electronic form (e.g., as a file in the Capella format). We distinguish between two kinds of note objects: *explicit* and *implicit* ones. For *explicit* objects all note parameters such as measure, beat, duration, and pitch are given explicitly. In view of the synchronization algorithm we only use the musical onset time and pitch. We represent each explicit note object by a tuple $(e, p) \in \mathbb{Q} \times [0 : 127]$, where $[a : b] := \{a, a + 1, \ldots, b\}$ for integers a and b. Here, we identify a pitch with the corresponding MIDI pitch given by an integer between 0 and 127. Furthermore, the musical onset time $e \in \mathbb{Q}$ is computed by $e = r \cdot (m - 1) + b$ if the piece of music has r beats per measure, where $m \in \mathbb{N}$ and $b \in \mathbb{Q}$ denote the measure and beat respectively of the explicit note object. For example, the first and fourth explicit note object in the right hand of the Aria (Fig. 3) are given by $(0, 79)$ and $(2.75, 83)$ respectively.

By an *implicit* note object we understand notes or a group of notes with an additional specification such as a trill, an arpeggio or grace notes. Implicit objects allow different realizations, depending on the epoch and the actual interpretation. To get this ambiguity under control we introduce the concept of a fuzzy note. A *fuzzy note* is defined to be a tuple (e, H) consisting of an onset time $e \in \mathbb{Q}$ and a set of alternative pitches $H \subset [0 : 127]$. Then an implicit note object, such as a trill, is represented by the musical onset time of a certain main

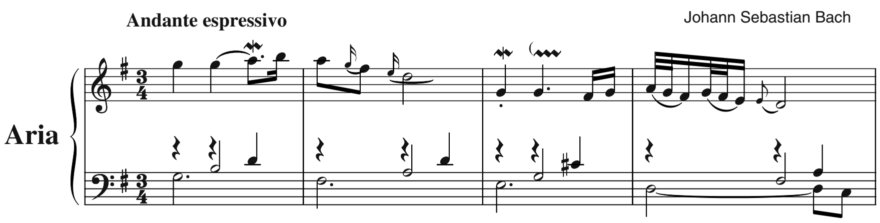


Fig. 3. First four measures of the *Aria con Variazioni* by J. S. Bach, BWV 988.

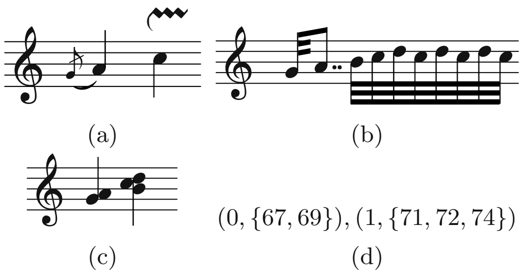


Fig. 4. Appoggiatura and trill, (a) notation, (b) possible realization, (c) fuzzy note, (d) encoding.

note and the set of all pitches appearing in a possible realization of this object (see Fig. 4 for an illustration). For example, the third note object of the first measure in the right hand of the Aria (Fig. 3) is given by $(3, \{79, 81\})$.

Using this encoding we may assume that a score is given by a subset $S \subset \mathbb{Q} \times 2^{[0:127]} \times 2^{[0:127]}$, where $2^{[0:127]}$ denotes the set of all subsets of $[0 : 127]$. Here, in a triple $(e, H_0, H_1) \in S$ the subset $H_0 \subset 2^{[0:127]}$ consists of all pitches of explicit note objects having musical onset time e and similarly the subset $H_1 \subset 2^{[0:127]}$ consists of all pitches of implicit note objects having musical onset time e.

We now turn to the data stream given in PCM-format. As described in Section 3, we extract a set of possible candidates of note objects given by their physical onset times and pitches (including in general erroneous objects). In view of the synchronization it is useful to further process this extracted data by quantizing the onset times. Simply speaking, we pool all note objects by suitably adjusting those physical onset times which only differ by some small value — e.g., smaller than a suitably chosen $\Delta > 0$ — since these note objects are likely to have the same musical onset time in the corresponding score format. After quantization we also may assume that the extracted PCM-data is given by a subset $P_\Delta \subset \mathbb{Q} \times 2^{[0:127]}$. Note that in the PCM-case there are only explicit note objects.

Altogether, we may assume that the score and the Δ-quantized extracted PCM-data are given by the sets

$$S = [(s_1, S_{01}, S_{11}), \dots, (s_s, S_{0s}, S_{1s})] \quad \text{and} \quad P_\Delta = [(p_1, P_{01}), \dots, (p_p, P_{0p})].$$

Here, the s_i, $1 \leq i \leq s$, denote the musical onset times and the p_j, $1 \leq j \leq p$, denote quantized physical onset times. Furthermore, $S_{0i}, S_{1i}, P_{0j} \subset [0 : 127]$ are the respective sets of pitches for the explicit and implicit objects.

On the basis of S and P_Δ we now accomplish the SP-synchronization. Since the score and the PCM-data represent the same piece of music, it is reasonable to assume $s_1 = p_1 = 0$ by possibly shifting the onset times. Now, the goal is to partially link the onset times $s_1, \dots, s_s$ to $p_1, \dots, p_p$ by maximizing the matches of the corresponding pitch sets. In the following, we formalize this approach.

Definition 1. *A* score-PCM-match *(SP-match) of S and P_Δ is defined to be a partial map $\mu: [1 : s] \to [1 : p]$, which is strictly monotonously increasing on its domain satisfying $(S_{0i} \cup S_{1i}) \cap P_{0\mu(i)} \neq \emptyset$ for all $i \in \mathrm{Domain}(\mu)$.*

This definition needs some explanations. The fact that objects in S or P_Δ may not have a counterpart in the other data stream is modeled by the requirement that μ is only a partial function and not a total one. The monotony of μ reflects the requirement of faithful timing: if a note in S precedes a second one this also should hold for the μ-images of these notes. Finally, the requirement $(S_{0i} \cup S_{1i}) \cap P_{0\mu(i)} \neq \emptyset$ prevents that onset times are linked which are completely unrelated with respect to their pitches.

Obviously, there are many possible SP-matches between S and P_Δ. By means of a suitable cost function we can compare different matches. The goal is then to compute an SP-match minimizing the cost function. To simplify the notation we identify the partial function μ with its graph $\mathrm{Graph}(\mu) := \{(i_1, j_1), \cdots, (i_\ell, j_\ell)\}$, where $\{i_1 < \cdots < i_\ell\} \subseteq [1 : s]$ and $\{j_1 = \mu(i_1) < \cdots < j_\ell = \mu(i_\ell)\} \subseteq [1 : p]$. In the following definition we assign costs to each SP-match μ. In this, we make use of a parameter vector $\pi := (\alpha, \beta, \gamma, \delta, \zeta, \Delta) \in \mathbb{R}^6_{\geq 0}$ consisting of six real parameters which will be specified later.

Definition 2. *With respect to the parameter vector $\pi := (\alpha, \beta, \gamma, \delta, \zeta, \Delta) \in \mathbb{R}^6_{\geq 0}$ the* SP-cost *of an SP-match μ between a score S and a Δ-quantized set P_Δ of the corresponding PCM-document is defined as*

$$\begin{aligned} C^{\mathrm{SP}}_\pi(\mu|S, P_\Delta) := {} & \alpha \cdot \sum_{(i,j)\in\mu} \Big(|S_{0i} \setminus P_{0j}| + \lambda(i,j) \Big) \\ & + \beta \cdot \sum_{(i,j)\in\mu} \Big(|P_{0j} \setminus (S_{0i} \cup S_{1i})| + \rho(i,j) \Big) \\ & + \gamma \cdot \sum_{k \notin \mathrm{Domain}(\mu)} \Big(|S_{0k}| + \sigma(k) \Big) + \delta \cdot \sum_{t \notin \mathrm{Image}(\mu)} |P_{0t}| \\ & + \zeta \cdot \sum_{(i,j)\in\mu} \Big| s_i - p_j \cdot \ell(S)/\ell(P) \Big|. \end{aligned}$$

This definition also requires some explanations. The sum corresponding to the factor α represents the cost of the non-matched explicit and implicit note objects of the score S. To be more accurate, the cardinality $|S_{0i} \setminus P_{0j}|$ measures the cost arising from the difference of the set of explicit note objects at the ith onset time of S and the set of explicit quantized note objects at the jth onset time of P_Δ. Furthermore, $\lambda(i,j)$ is defined to be 1 if and only if the score S has an implicit note object at the ith onset time and P_Δ has no counterpart at the jth onset time, i.e., $S_{1i} \neq \emptyset$ and $S_{1i} \cap P_{0j} = \emptyset$. In all other cases $\lambda(i,j)$ is defined to be 0. Next, we consider the sum corresponding to the factor β. The first summand in the brackets measures the cost of (possibly erroneously) extracted notes at the jth physical onset time whose pitches do not lie in $S_{0i} \cup S_{1i}$. Furthermore, $\rho(i,j)$ is defined to be $|P_{0j} \cap S_{1i}| - 1$ if $P_{0j} \cap S_{1i} \neq \emptyset$. Otherwise $\rho(i,j)$ is defined to be 0. In other words, for the implicit note objects only *one* match is free of cost whereas each additional match is penalized. (This is motivated by the idea that all notes belonging to a realization of a fuzzy note are expected to have pairwise distinct physical onset times.) The sum corresponding to γ accounts for all onset times of the score which do not belong to the match μ. The first term within the brackets counts the number of explicit note objects at the kth onset time, $k \notin \mathrm{Domain}(\mu)$. Furthermore, $\sigma(k)$ is defined to be 1 if there is a non-matched implicit note object and 0 if there is no implicit note object at the kth onset time. (The idea is that a non-matched fuzzy note should only be penalized by 1 since it only represents a set of alternatives.) The sum corresponding to δ accounts for the cost of those notes in P_Δ which do not have a counterpart in S. Finally, the last sum corresponding to ζ measures an adjusted ℓ^1-distance (also known as Manhattan-distance) of the vector pairs $(s_i, p_j)_{(i,j)\in\mu}$, where $\ell(S)$ and $\ell(P)$ denote the differences of the last and the first musical respectively physical onset times (a kind of musical and physical duration). By this sum one penalizes matches with large relative time deviations thus preventing large global deviations in the synchronization.

In the following we fix a parameter vector π, a preprocessed score S, and quantized extracted PCM-data P_Δ. Note that if μ is an SP-match then also $\mu' := \mu \setminus \{(i,j)\}$ for some $(i,j) \in \mu$. An easy computation shows

$$\begin{aligned} C_\pi^{\mathrm{SP}}(\mu|S,P_\Delta) - C_\pi^{\mathrm{SP}}(\mu'|S,P_\Delta) = \alpha &\cdot \Big(|S_{0i} \setminus P_{0j}| + \lambda(i,j) \Big) \\ +\beta &\cdot \Big(|P_{0j} \setminus (S_{0i} \cup S_{1i})| + \rho(i,j) \Big) \\ -\gamma &\cdot \Big(|S_{0i}| + \sigma(i) \Big) - \delta \cdot |P_{0j}| \\ -\zeta &\cdot \Big| s_i - p_j \cdot \ell(S)/\ell(P) \Big| \end{aligned} \tag{1}$$

Now, one can determine a cost-minimizing SP-match by means of dynamic programming. We recursively define a matrix $C = (c_{ij})$ with $i \in [0:s]$ and $j \in [0:p]$. First, initialize $c_{0j} := c_{i0} := C_\pi^{\mathrm{SP}}(\emptyset|S,P_\Delta)$ for all $i \in [0:s], j \in [0:p]$. Note that this accounts for the costs that there is no match at all between S and P_Δ. At position $(i,j) \in [1:s] \times [1:p]$ the value c_{ij} expresses the cost for a cost-minimizing SP-match within the subset $[1:i] \times [1:j] \subset [1:s] \times [1:p]$. Hence,

c_{sp} expresses the minimal cost of a global SP-match. For $(i,j) \in [1:s] \times [1:p]$, the value c_{ij} is defined as

$$c_{ij} := \min\{c_{i,j-1}, c_{i-1,j}, c_{i-1,j-1} + d_{ij}^{\mathrm{SP}}\},$$

where

$$d_{ij}^{\mathrm{SP}} := \begin{cases} \text{right hand side of Eq. (1)}, & \text{if} \quad (S_{0i} \cup S_{1i}) \cap P_{0j} \neq \emptyset, \\ 0, & \text{otherwise.} \end{cases}$$

Using the resulting matrix $C = (c_{ij})$, the following algorithm computes a cost-minimizing SP-match:

```
SCORE-PCM-SYNCHRONIZATION($C$,$s$,$p$)
1   $i := s$, $j := p$, SP-Match $:= \emptyset$
2   while $(i > 0)$ and $(j > 0)$
3      do if $c[i,j] = c[i,j-1]$
4            then $j := j-1$
5         else if $c[i,j] = c[i-1,j]$
6            then $i := i-1$
7         else SP-Match $:=$ SP-Match $\cup \{(i,j)\}$, $i := i-1$, $j := j-1$
8   return SP-Match
```

In the next section we give an example to illustrate this procedure and report some of our experiments. As we mentioned before, SM- and MP-synchronization can be done similarly to the SP-case. Furthermore, other synchronization problems such as synchronization of two PCM-data streams P_1 and P_2 ($\mathrm{P}_1\mathrm{P}_2$-synchronization) may be achieved by using a score S as a reference and carrying out both an SP_1- and an SP_2-synchronization.

We conclude this section with some comments on the parameter vector $\pi := (\alpha, \beta, \gamma, \delta, \zeta, \Delta)$. In most of our experiments we set the quantization constant to $\Delta = 50$ ms. This threshold was chosen since it represents a good compromise between psychoacoustically distinguishable asynchronisms of chords and the shortest possible musical note durations. By the parameters α and β one can weight the cost for the symmetric difference of pitch sets corresponding to matched onset times, whereas by the parameters γ and δ one can weight the cost of those note objects which do not have a counterpart in the other data stream. One meaningful standard choice of the parameters is $\alpha = \beta = \gamma = \delta = 1$. However, if one wants to penalize non-matched onset times, for example, one may increase γ and δ. In the case $\zeta = 0$ the last sum of the cost function remains unconsidered. Increasing ζ will hamper matches (i,j) whose onset times s_i and p_j differ too much with respect to their relative positions in their respective data stream. In other words, excessive global time divergence in the synchronization of the two data streams can be controlled.

5 An Example

We illustrate the SP-synchronization by means of the example from Fig. 3. The score S represents the first four measures of the Aria. The PCM-version P represents a recording or the same measure performed on a Steinway grand piano. The physical length is $\ell(P) = 13$ sec. The quantization constant is set to $\Delta = 50$ ms. In the following, we restrict ourselves to the second measure, where two appoggiaturas appear in the right hand of the score. Those are modeled by fuzzy notes. Table 1 shows the note objects of the score S and the Δ-quantized extracted PCM-data P_Δ.

S				P_Δ		
i	s_i	S_{0i}	S_{1i}	j	p_j	P_{0j}
5	3	{54, 81}	∅	7	3.86	{54, 81}
6	3.5	∅	{78, 79}	8	4.47	{79}
				9	4.75	{66, 78}
7	4	{57}	{74, 76}	10	5.06	{57, 66, 76}
				11	5.71	{57, 74}
8	5	{62}	∅	12	6.39	{57, 62}

Table 1. Note objects for the score S and the Δ-quantized extracted PCM-data P_Δ for the second measure of the aria (cf. Fig. 3).

This example also illustrates two typical phenomena appearing in the extracted PCM-data. Firstly, in position $j = 10$ the extracted note of pitch 57 also appears in positions 11 and 12. This can be explained as follows: this note continues to sound and, at every new note attack (e.g., note of pitch 74 at position 11 or note of pitch 62 at position 12), the extraction algorithm again interprets the note of pitch 57 as a "new" note. Secondly, in position 9 appears an "erroneously" extracted note of pitch 66 which differs from the expected note of pitch 78 by an octave. This might be caused by the harmonics of the still sounding note of pitch 54 at position 7 and the new note of pitch 78 at position 9. Actually, these "octave errors" are typical for the extraction algorithm. To tackle this problem one can restrict oneself to only considering pitches which are reduced modulo 12 when using the note parameters as input for the synchronization algorithm. In spite of this reduction one is still left with sufficient information for a successful synchronization.

Table 2 shows the part of the cost matrix $C = (c_{ij})$ corresponding to the second movement using the parameter vector $\pi = (1, 1, 1, 1, 22, 50)$ and a modulo 12 reduction of the pitches.

From C one can compute the global SP-match μ. For the second measure this gives the matches $\{(5, 7), (6, 8), (7, 10), (8, 12)\}$ which are also printed in bold face in the above table. Note that the SP-algorithm has matched for both appoggiaturas of the score S the corresponding fuzzy notes at position $i = 6$ and

	6	7	8	9	10	11	12
4	114.8476	114.8476	114.8476	114.8476	114.8476	114.8476	114.8476
5	114.8476	**111.8700**	111.8700	111.8700	111.8700	111.8700	111.8700
6	114.8476	111.8700	**110.8132**	110.8132	110.8132	110.8132	110.8132
7	114.8476	111.8700	110.8132	110.8132	**107.4704**	107.4704	107.4704
8	114.8476	111.8700	110.8132	110.8132	107.4704	107.4704	**106.7956**

Table 2. Excerpt of the cost matrix $C = (c_{ij})$ corresponding to the second movement of the aria using the parameter vector $\pi = (1, 1, 1, 1, 22, 50)$ and a modulo 12 reduction of the pitches.

$i = 7$ respectively with the corresponding first notes of the appoggiaturas of P_Δ at position $j = 8$ and $j = 10$ respectively.

6 Experiments

We have implemented prototypes of the extraction algorithms described in Section 3 and the synchronization algorithms using MATLAB. Our algorithms for SM-, SP-, and MP-synchronization have been evaluated on a variety of classical polyphonic piano pieces of different complexity and length (ranging from 10 to 60 seconds) played on various instruments. For handling MIDI files within the MATLAB environment, we have implemented several tools for importing, exporting and visualizing MIDI data. Those tools have been made available for the research community, see [8]. Furthermore, we have systematically generated a library of more than one hundred test pieces both in MIDI- and PCM-format played on a MIDI-piano, a Steinway grand piano, and a Schimmel piano. In some of those pieces our player has deliberately built in excessive accelerandi, ritartandi, rhythmic distortions, and wrong notes. Even in these extreme situations, where one unsurprisingly has many “erroneously” extracted note objects which considerably differ from the score-data, our SP-synchronization algorithm resulted in good overall global matches which are sufficient for most applications mentioned in the introduction. Even more, in case of rather accurately extracted note parameters our synchronization algorithms could resolve subtle local time variations in an interpreted version of the piano piece.

As has already been observed in previous work, the evaluation of synchronization methods is not straightforward and requires special care. First, one has to specify the granularity of the alignment which very much depends on the application in mind. For example, if one is interested in a system which simultaneously highlights the current measure of the score while playing a corresponding interpretation (as a reading aid for the listener), a deviation of the alignment of a note or even several notes might be tolerable. However, for musical studies or when used as training data for statistical methods a synchronization at note-level or even ADSR-level (Attack-Decay-Sustain-Release) might be required.

Intuitive objective measures of synchronization quality may be the percentage of note events which are correctly matched, the percentage of mismatched notes, or the deviation between the computed and optimal tempo curve. (The output of a synchronization algorithm may be regarded as a tempo deviation or *tempo curve* between the two input data streams.) However, such a measure will fail if the note events which should be aligned do not exactly correspond (such as for trills, arpeggios, or wrong notes). In such a case the measure might give a low grade (bad score) which is not due to the performance of the algorithm but due to the quality of the input streams. Here, one would rate a synchronization as "good" if the musically most important note events are aligned correctly. Unfortunately, to determine such musically important note events a manual interaction is required, which makes the procedure unfeasible for large-scale examinations. Similarly, the measurement of tempo curves requires some ground truth about the desired outcome of the synchronization procedure. Moreover, if a synchronization algorithm is intended to handle even changes in the global structure of the musical piece such as an additional chorus or a missing bridge [18], modified notions of tempo curves have to be considered.

For those reasons, we decided to evaluate our synchronization results mainly via sonification, which allows an acoustic assessment on the ADSR-level. The design of suitable objective measures, which allow a systematic and automatic assessment of the synchronization results, is still an open research problem.

In the following we describe one of our experiments where we started with an uninterpreted score-like MIDI-version and an interpreted PCM-version of a given piano piece. Using the results of our MP-synchronization, we automatically modified the onset times of the MIDI-data stream to correspond to the PCM-data stream with regard to the global tempo and the local tempo variations. This results in an "expressive" MIDI-version which represents a sonification of our synchronization results. In case of good extraction parameters the modified MIDI-version rhythmically sounds like a real interpretation of the underlying piano piece.

To demonstrate our synchronization results we made some of our material available at `http://www-mmdb.iai.uni-bonn.de/download/syncDemo/`, where we provide the underlying test material as well as synchronized versions for several excerpts of classical piano pieces. For each example we supply the following data:

(1) An uninterpreted MIDI-version (representing the score of the underlying piece of music).
(2) An interpretation the score (1) performed on an acoustic grand piano (WAV format).
(3) A MIDI-file containing the score parameters (onset times and pitches) extracted from (2) using our extraction algorithm.
(4) An expressive MIDI-version created from (1) by modifying the onset times according to our synchronization result. (The note lengths are adopted from (1). Furthermore, the onset times of non-matched note events of (1) where modified via a simple interpolation.)

(5) An audio-file (WAV format) containing a mix of the the original interpretation (2) in the right audio channel and a synthesized version of the expressive MIDI (4) in the left audio channel.

In some of the cases, two different synchronized versions are given, which have been derived using slightly different choices of the parameter vector. As can be observed by listening to the audio-files (5), the synchronization results for the Mozart and the two Bach variations are very accurate. In the Burgmüller example, our player deliberately built in excessive local tempo deviations. Even in this case our algorithm computed a good global alignment with occasional local inaccuracies. The Bach Aria is synchronized very well except for the trill. This can be explained as follows: The number of notes in the trill of MIDI-version (1) is larger than the one of the corresponding trill in the interpreted PCM-version (2). In other words, there is no exact correspondence of note events in the data streams to be aligned (just as in the case where one has wrong notes in the interpretation). In this case, all what one can expect is not an alignment on the note-level but a rough alignment of the whole regions corresponding to the trills. This is actually what our algorithm does: the notes before and after the trill are matched correctly and the additional notes within the trill of the MIDI-version are not matched at all — only for the sonification we determined the onset times of the additional notes by interpolation which does not reflect the actual synchronization result. This also shows that the method of sonification has to be treated with care when evaluating synchronization results.

The last example is also illustrated by Fig. 5 corresponding to the first four measures of the of the Bach Aria BWV 988. The top part of the figure shows the piano roll representation of the score-like MIDI-version, the medium part shows the piano roll representation of the extracted note parameters from our interpreted PCM-version (note lengths are not considered), and the bottom part shows the piano roll representation of the "expressive" MIDI-version representing the synchronization result (note lengths of the score-like MIDI-version are used). As one can see in the medium part, the extraction algorithm also generates erroneous score-parameters such as typical octave errors coming from the harmonics. The quality of the extracted note parameters may also be low in parts consisting of many short notes. Furthermore, the uninterpreted MIDI-data stream and the extracted note parameters do note match well around the trill passage. Nevertheless, using the concept of partial matches we do not attempt to force alignment of non matching note objects. The bottom part of Fig. 5 shows that in spite of the minor extraction quality, this strategy allows us to find a proper MP-synchronization.

7 Conclusions

In this paper we have proposed algorithms for the automatic synchronization of different versions of a polyphonic piano piece given in different data formats (score, MIDI, PCM). Our implementation yields good synchronization results

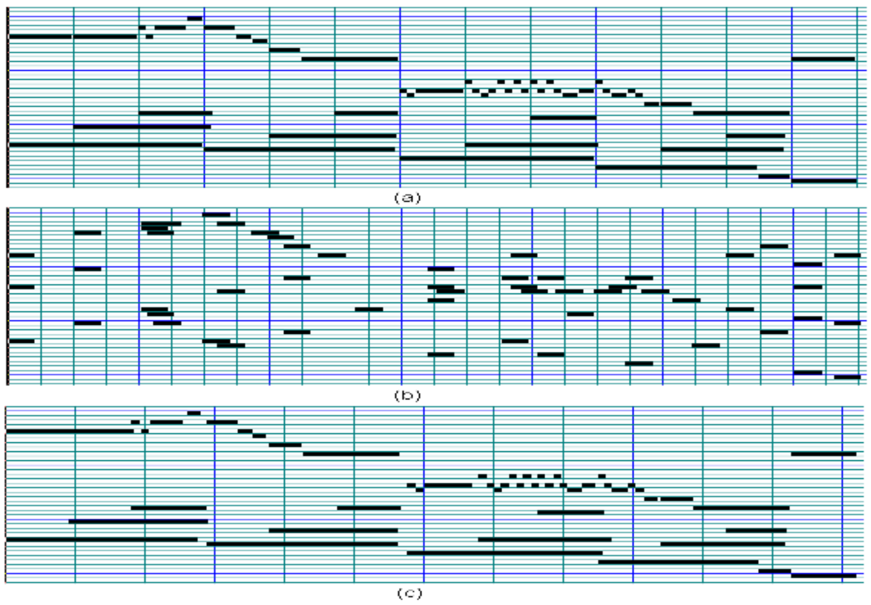


Fig. 5. (a) Uninterpreted MIDI of the first four measures of the Bach Aria BWV 988, (b) MIDI representation of the extracted data from an interpreted version, and (c) expressive MIDI version representing the synchronization results.

even for complex PCM-based polyphonic piano CD-recordings. One of the decisive features is a carefully designed cost function which not only penalizes non- or partially matched note objects but also large relative global time deviations. The parameter vector π allows to weight different aspects in the matching process and leaves room for experiments. In contrast to related approaches [17,18], our method fully works on the symbolic, i.e., note-level domain. Working with such score-like features not only accounts for high accuracy — even when having strong local time deviations — but also makes DWT-like computation feasible due the relatively small data sets used in the actual synchronization. A further qualitative advantage of our matching-strategy is the focus on *partial matches.* Here we only include promising score-based matches in the final synchronization result rather than forcing a match for each note object in each of the two versions to be aligned.

Our system works off-line, where the computational bottle-neck lies in the preprocessing step needed to extract note-parameters from the PCM-files (where we up to now did not use any score information so far). An ongoing research project is to exploit the score information already in the extraction step. (See also [16] for a similar approach.) This prior knowledge allows to use prediction methods (in particular Kalman-filtering) which in connection with time-varying comb filters may result in extraction algorithms running in real-time. Such fast algorithms may be at the expense of possible lower quality of the extraction parameters. However, even lower quality of the parameters may be sufficient for a successful synchronization when using a suitably designed cost function which is robust under erroneous parameters.

Future work will also be concerned with properly defining various different types of synchronization problems some of which have been sketched in the last section. Based on such clearly defined problems, suitable measures for comparing synchronization results and assessing their overall quality have to be devised.

Automatic music processing is extremely difficult due to the complexity and diversity of music data. One generally has to account for various aspects such as the data format (e.g., score, MIDI, PCM), the genre (e.g., pop music, classical music, jazz), the instrumentation (e.g., orchestra, piano, drums, voice), and many other parameters (e.g., dynamics, tempo, or timbre). Therefore for most problems in computational musicology there are no universal algorithms yielding optimal solutions for all kinds of music. The approach by Turetsky et al. [18] works for general instrumentations but has weaknesses concerning large local tempo variations, whereas our approach captures even large local time deviations of a specific interpretation but is limited concerning the instrumentation. The approach of Soulez et al. [17] may be classified in between the former two. For the future it seems to be promising to build up a system which incorporates different, competing strategies (instead of relying on one single strategy) in combination with statistical methods as well as explicit instrument models in order to cope with the richness and variety of music.

References

1. Arifi, V.: *Algorithmen zur Synchronisation von Musikdaten im Paritur-, MIDI- und PCM-Format.* PhD thesis, Universität Bonn, Institut für Informatik, 2002. http://hss.ulb.uni-bonn.de:90/ulb_bonn/diss_online/math_nat_fak/2002/arifi_vlora
2. Bobrek, M., Koch, D.: *Music Signal Segmentation Using Tree-Structured Filter Banks.* Journal of Audio Engineering Society, Vol. 46, No. 5, May 1998.
3. Dannenberg, R. B.: *An On-Line Algorithm for Real-Time Accompaniment.* Proceedings of ICMC, 1984.
4. Dannenberg, R. B., Mukaino, H.: *New Techniques for Enhanced Quality of Computer Accompaniment.* Proceedings of ICMC, 1988.
5. Desain, P., Honing, H., Heijink, H.: *Robust Score-Performance Matching: Taking Advantage of Structural Information.* ICMC Proceedings 1997, pp. 377-340.
6. Foote, J.: *Automatic Audio Segmentation Using a Measure of Audio Novelity.* Proceedings of IEEE International Conference on Multimedia and Expo, vol. I, 2000, pp. 452-455.
7. Foster, S., Schloss, W.A., Rockmore, A.J.: *Towards an Intelligent Editor of Digital Audio: Signal Processing Methods* Computer Music Journal, Vol. 6, No. 1, Spring 1982.
8. Kurth, F.: *A Matlab Toolbox for Handling Standard MIDI files.* `http://www-mmdb.iai.uni-bonn.de/download/miditools/`, Department of Computer Science III, Bonn University, Germany, 2003.
9. Large, E. W.: *Dynamic programming for the analysis of serial behaviours.* Behaviour Research Methods, Instruments, & Computers. 1993.

10. Klapuri, A., Virtanen, T., Holm, J.: *Robust Multipitch Estimation for the Analysis and Manipulation of Polyphonic Musical Signals.* In Proc. COST-G6 Conference of Digital Audio Effects, DAFx-00, Verona, Italy, 2000.
11. Mazzola, G.: *The Topos of Music.* Birkhäuser Verlag, 2002.
12. Orio, N., Lemouton, S., Schwarz, D.: *Score Following: State of the Art and New Developments*, Proc. of the Conference of New Interfaces for Musical Expression NIME, Montreal, 2003.
13. Rabiner, L.R., Juang, B.-H.: *Fundamentals of Speech Recognition,* Englewood Cliffs, Prentice Hall, 1993.
14. Raphael, C.: *Automatic Segmentation of Acoustic Musical Signals Using Hidden Markov Models.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 4, April 1999.
15. Raphael, C.: *Automatic Transcription of Piano Music.* ISMIR 2002, IRCAM, Paris, Conference Proceedings, 2002.
16. Scheirer, E. D.: *Extracting Expressive Performance Information from Recorded Music.* Unpublished M. S. thesis, MIT Media Laboratory, 1995 http://web.media.mit.edu/ eds/thesis.pdf
17. Soulez, F., Rodet, X., Schwarz, D: *Improving polyphonic and poly-instrumental music to score alignment.* International Conference on Music Information Retrieval, Baltimore, 2003.
18. Turetsky, R. J., Ellis, D. P., *Force-Aligning MIDI Syntheses for Polyphonic Music Transcription Generation.* International Conference on Music Information Retrieval, Baltimore, USA, 2003.

Towards an Intelligent Score Following System: Handling of Mistakes and Jumps Encountered During Piano Practicing

Mevlut Evren Tekin, Christina Anagnostopoulou, Yo Tomita

Sonic Arts Research Centre,
Queen's University of Belfast,
BT7 1NN, Belfast, United Kingdom
{e.tekin, c.anagnostopoulou, y.tomita}@qub.ac.uk

Abstract. Score following has been an important area of research in AI and music since the mid 80's. Various systems were developed, but they were predominantly for providing automated accompaniment to live concert performances, dealing mostly with issues relating to pitch detection and identification of embellished melodies. They have a big potential in the area of education where student performers benefit in practice situations. Current accompaniment systems are not designed to deal with errors that may occur during practising. In this paper we present a system developed to provide accompaniment for students practising at home. First a survey of score following will be given. Then the capabilities of the system will be explained, and the results from the first experiments of the monophonic score following system will be presented.

1 Introduction

A score following system monitors a live performance mainly for the purpose of providing an automatic accompaniment to live performers. The first examples of score following systems were developed by Dannenberg [4] and Vercoe [15] in two different studies and were capable of handling the input from a monophonic sound source. Subsequent score followers were polyphonic and had extended capabilities to handle ornaments such as trills and mordents as well as more widely dispersed pitch clusters such as glissandos [1].

Although score following systems were originally invented for the live performances, they also have potential for providing the core function for tutoring systems and performance evaluation tools. Based on his score following systems, Dannenberg [7] has developed the Piano Tutor, providing guidance to students. The Piano Tutor is an expert system that uses score following techniques to monitor student progress and identify weaknesses. When the students successfully complete a set of exercises, the expert system then advances on to a higher level of exercises. Similarly, Bora's [2] system is also built to evaluate student performances, although it does not provide any further tutoring driven by an expert system [2].

All the above systems rely on MIDI signals as input and they use string matching methods. More recently, new systems have been developed that have the capability to handle audio input captured by microphone: they use stochastic pitch detection algorithms when receiving the input from their sound sources [3], [11], [12], [13]. Their main strength, on the one hand, is the ability to follow the performances that do not depend on the fixed pitches. On the other hand, they are incapable of handling even the shortest of jumps in the performance, which is the main weakness. They also need to be trained through several rehearsal sessions in order for them to be able to adjust the state transition weights to follow the performance precisely.

MuseBook Score, which is marketed as an automatic page-turning system for pianists, is another score following system. It follows the performance by listening to pitch. According to AMuseTec website [16], the system is capable of following the performance and displaying the current note being played by the pianist. However, the system does not offer any accompaniment features in live performance. They acknowledge that it is due to the delay caused during the pitch detection process. Although there is some video demonstration of the running system at AMuseTec website, there is no published information about the techniques used in the product.

The idea for score following systems originates from a very successful product called “Music Minus One” (MMO), which is in short a special recording of musical pieces (piano concertos and violin sonatas are two of the most popular genres) made for an instrumentalist who does not have accompanists during the practice sessions [13]. Normally MMO comes with two versions of the same piece, the one complete performance serving as a guide as to how the ensemble should be worked out, and the other the accompaniment only. It assumes that the performer of the featured instrument plays the MMO recording on the stereo and plays along with the second version of recording, trying to fill in the solo part that is deliberately omitted from the recording. While the concept is excellent, its limitations are obvious. Firstly, MMO does not take into account the fact that the learners cannot perform the piece at the set tempo from the outset; students may need to learn to play slowly first. Secondly, it does not allow the student to stop, make mistakes and repeat challenging sections.

Our score following system is designed to be used by students learning the pieces on the piano. In concert performances the number of errors will be very small and jumps are not normally expected. Based on this assumption, most score following systems that are developed for concert performances focus on the problems related to pitch detection and the identification of ornaments. Our approach deals with significant jumps in the score, repetition, in addition to the sudden changes of tempo, wrong pitches and identification of ornaments that had previously been achieved with some degree of success by the previous scholars in the field.

2 The System

In this paper we describe our system, which is aimed primarily at students engaged in practicing the piano. Its main objective is to facilitate their practice by providing a reliable accompaniment to the students learning at home environment. The system should be able to cope with a large number of mistakes: not only pitch or rhythm but also jumps, repeats and sudden tempo changes that are made deliberately or unconsciously during the practice. It may be worth adding that –ornaments especially those of technically demanding ones– are often played erratically, and for this reason students might be advised to omit them during the early stages of practicing. The system should be designed to tolerate such interpretative problems.

2.1 Input/Output

For the present work we selected a Yamaha Disklavier Pro series piano, which has a MIDI input and output. The student pianist (user) plays the piece on the piano, and the system provides the accompaniment, which is also played on the same instrument. The system can provide the suitable accompaniment for left hand and right hand practices. The user can also choose between two different tempo options, which will be explained in Section 2.4.

2.2 The Score Following Algorithm

Common Practice Notation (CPN) is a universal notational system representing music in written form. Because it reduces the principal dimensions of music to pitch and duration, it represents music in a very compact way. The continuous musical stream is quantized into discrete notes, resulting in losing some information originally present in music. However, this is not a problem as part of the lost information about the music can be reconstituted through the 'interpretation' of trained performers. To help the performers recover this lost information, CPN also makes use of additional expression marks. Thus although the basic unit representation is heavily quantized, different performers can give different performances by interpreting these notational symbols and instructions [8].

Unlike CPN, MIDI does not deal with any expression marks and instructions which are open to interpretation. Instead, all these are explicitly spelt out in the MIDI score (such as trills and crescendos). However, in actual performances, ornaments are seldom played identically, even by the same person. Therefore, it would be impractical to encode a single interpretation of an ornament in the MIDI file as it will deny the player any licence to have his or her performance style. For this reason, the MIDI files the system uses do not include any ornaments, allowing the players to add their own interpretation to the performance. The way the system handles ornaments is discussed in section 2.3.

In our system, the music is represented as a series of events. The score to be followed is represented by the tuple *(pitch,duration)* in every event. To each event a list of accompaniments is attached. This list contains pitch, duration and note_on/note_off messages for the accompaniment to be played within the corresponding event.

The system is a multi-agent system, where agents represent single events of the score representation. Each agent is sequentially linked to the next one in a manner of a linked list. We call this linked list the *score array*. Each agent holds the pitch information and continuously monitors the performance. When a student plays a note, all agents corresponding to the pitch being played are activated to their maximum activation values. When a key is released, the previously-activated agents are deactivated. But the activation values of the deactivated agents are not assigned to the value of zero; instead, the activation values decrease slowly until they reach zero. In a musical composition, the same notes are used many times, hence many agents being activated at a time. But as the student plays along the piece, a region with high activity appears in the score array. The agent having the highest activation value is picked up the as the score location by the system.

There are two different strategies for decreasing the activation values. In the first one, the activation values are decreased every time the user plays a new note. In the second one the activation values are decreased at user defined time intervals (200 milliseconds – 1000 milliseconds). The first option acts like a "pause" key. If the user stops and does not play for a long time, the system resumes playing from the last position when the user starts playing again. However, if the second option is selected, the player has to play continuously or otherwise, the system will reset itself. The first option is more successful at responding to repeats while the second one can pick up new start locations better. The second option is more suitable for concert ready performances.

The communication between the computer and the piano is provided by a USB MIDI interface which has the unidirectional delay of 110 milliseconds. Because of this communication delay, we had to skip a validation step which was originally incorporated in our score following algorithm. In the original algorithm, the system first checked the key pressed by the player before sending the accompaniment data to the piano. If the pressed key was identical with the expected pitch, the system sent the appropriate accompaniment to the piano. However, due to the delay, the accompaniment was heard approximately 230 milliseconds too late. For this reason, in the current system as soon as the next expected note is computed by the system, without carrying out any validation process the accompaniment is sent 110 milliseconds before the player is expected to hit the key.

The system makes use of a *confidence mechanism* which acts like a short term memory. This mechanism enables the system to reliably follow a piece with number of repetitions by reducing the search space. The initial experiments without the confidence mechanism were carried out with pieces by Beethoven, Mozart and J.S. Bach. The music was played with one hand only on the piano and the system was assigned to provide the other hand accompaniment. Although the system performance was satisfactory with Beethoven and Mozart, it encountered some difficulties with the music of Bach. One of the pieces we used was the Prelude in C minor from *the Well-Tempered Clavier, Book I* (BWV 847/1) which is based on the broken-chord motif in various keys. It would seem that the repetition of the identical melodic patterns occasionally caused the system to lose the score position even if the player was not making a jump or an error.

Repetition is an inherent property of music. Many traditional musical forms contain sections that repeat the materials used earlier in the piece. For example, a traditional Rondo would be expected to have a structure of A-B-A-C-A (and so forth), where all As are usually exact repetitions in the same key. For our score following system, this may create an ambiguity. This problem has been resolved by including a confidence mechanism, which increases the weight of a specific section when its prior section is followed successfully. So, for example, when section B is being played, the weight of the second A section is then increased. But when the player plays A, the system picks one of the A sections randomly as it would not have sufficient clues to determine as the weight of all A sections would be the same in this case. As soon as the player starts playing the next section—either B or C—the system can then decide within a single note which A section it played.

The confidence mechanism artificially increases the weight of the most active region on the score array so that the competing hypotheses—i.e. other possible candidate locations like as given in the previous example—would not be activated as a jump destination. To allow the score follower to react to a jump, this active region with increased weight receives heavier penalties—i.e. the active region receives an instruction to decay its strength faster than the competing hypotheses—than the other possible candidate locations when the player makes a mistake or jumps to another location of the score.

2.3 Handling of Trills and Ornaments

An important issue in performance is ornamentation. It often varies from edition to edition, and different performers may choose different ways of interpreting and executing ornaments. Score following systems should be able to cope with variations in the interpretation.

Dannenberg [5] uses a preprocessor to handle the problems caused by trills and ornaments. The preprocessor which contains a finite state machine is responsible from listening to the input and has two internal stages called "normal" and "trill/glissando". The score follower, which is referred as the matcher, receives the performance data from the preprocessor. When the preprocessor detects a special signal in the score like a trill or glissando, it changes its internal state to "trill/glissando" and stops sending any data to the matcher. The preprocessor does not change its state to normal and start sending data to the matcher until it receives a longer note or the next note after the trill/glissando is performed near its expected time.

In our approach, as mentioned above, all ornaments are omitted from the stored data representation of the score. This allows students to add their own ornaments to the musical piece; at the same time, it adds greater flexibility by allowing the user to skip ornaments, or play them incorrectly.

The system employs two heuristics in order to be able to identify the ornaments performed by the performer:

- the notes, performed unexpectedly, are around the expected key
- the ornament should end with the expected note

These heuristics deliver successful results in case of unexpectedly performed ornaments (Section 3.3).

2.4 Tempo

One of the most significant issues in autonomous accompaniment systems is the process of beat tracking. The system requires a constant notion of tempo to be able to play in synchronization with the performer. Although in a well-polished performance some tempo variations are frequently introduced by players, during the practice sessions tempo variations often occur at the technically challenging passages, and can be sudden and substantial.

The system implements two contrasting methods of handling the tempo variations: (1) user dependent tempo and (2) forced-tempo. The user has to choose one of these options at the beginning of the practice session.

When practising a piece, some students might play the technically demanding passages slower. In user-dependent tempo, the system instantly adjusts the tempo if the player makes a sudden change.

If the forced-tempo option is selected, the system does not change its tempo suddenly when detecting a sudden tempo change, but it adopts the new tempo gradually. This behaviour forces the student to catch up with the accompaniment. If the tempo changes are minor, caused for example by expressive performance, the system will change its tempo accordingly. The forced tempo option can be useful for the students who feel comfortable with the performance of the musical material and need to push the practised piece to the expected final speed.

To calculate the tempo changes in both cases, weighted averaging is used. In user dependent tempo, the last tempo changes have a higher weight than the previous ones. However, in forced tempo, although the last tempo changes also have a relatively higher weight, the weights are not as distinct as the ones in user dependent tempo.

3 Results

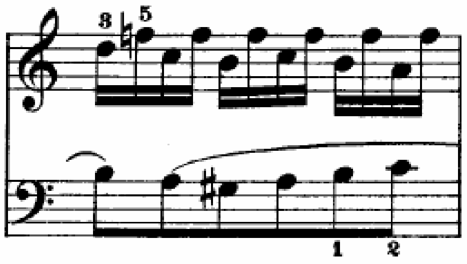
To illustrate the system's performance, we use the Prelude from English Suite in A minor (BWV 807) by J. S. Bach. While the player performed only the right hand part of the piece, the system accompanied the player with the left hand part. Although the right hand part is mostly monophonic, there are a few chordal passages. To be able to run the experiment we edited the piece and performed only the top voice in these passages. The results presented here make use of the confidence mechanism. The activation values are decreased every time the player presses a new key on the piano.

3.1 Experiment 1: Performance Without Mistakes

In the first experiment the player simply played through the whole piece from the beginning without making any errors or jumps. The system was able to accompany the player from the beginning to the end without making any mistakes.

3.2 Experiment 2: Performance with Wrong Notes and Jumps

Repeats are the most common type of jumps that one could make during the practice. To be able to test the reaction of the system in case such repeats occur, we prepared a scenario with some sections repeated over and over. These repeats contained some missing/wrong notes with their number decreasing in each repeat.



score performed	d f c f b f c♯ f d f c f b f c d d f c f b f c f b f a f
expected accompaniment	b - a - g♯ - - - b - a - g♯ · a - b - a - g♯ - a - b - c -
system response	- - a - g♯ - a - - - a - g♯ · a - - - b - g♯ - a - b - c -

I **II** **III** **IV**

Fig. 1. Excerpt from the Prelude from the *English Suite in A minor* (BWV 807) by J. S. Bach and the results from experiment 2. The highlighted regions marked I, II, III and IV show the difference between the expected accompaniment and the actual system response

Fig. 1 shows the actual score, score performed, expected accompaniment and system response of an example scenario of the practice described above. In this scenario, the student practises this particular bar over and over again until she or he can play the passage without mistakes. This example actually records that this student is making fewer mistakes at each repeat. The regions marked on the table highlight four areas of mismatches between the expected accompaniment and system response, which we discuss below.

When the player starts playing from a random point in the score, the system needs to 'listen' for a few notes in order to find the score location. During this time the system is not be able to provide any accompaniment (Region I, Fig. 1).

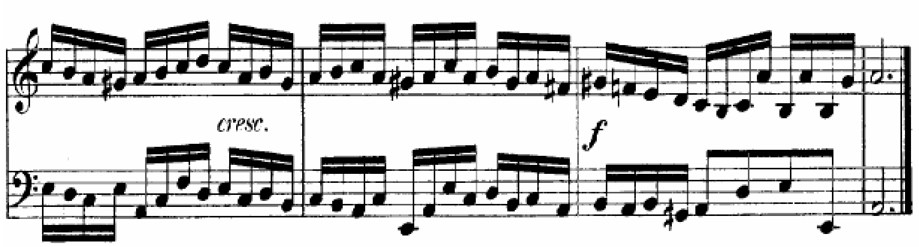
Region II shows another mismatch. The student here made a mistake, adding a sharp to the **c** of the score. Although the expected accompaniment would be silent, the system actually plays **a**, which would be the accompaniment to the correct **c** natural note. This is because the confidence mechanism tries to tolerate the error made by the player here and continues to provide an accompaniment despite the error.

The mismatch in Region III is caused by the jump performed by the player. The system requires sufficient clues, i.e. series of pitches, to be able to determine the position on the score.

In Region IV, the player starts repeating the section again. The system starts responding with a wrong note (**b** instead of **a**) after resuming the accompaniment. However the system resumes playing the correct accompaniment within two extra notes.

3.3 Experiment 3: Handling the Trills

In the third and last experiment, we tested the system's abilities to handle ornaments and trills by adding some trills which were not defined in the score. The results of this experiment are summarised in Fig. 2. Although the trills were not indicated in the stored data representation, the system was able to continue to provide an acceptable accompaniment, skipping a single note only (highlighted with label I in Fig. 2). Though trills are detected as wrong notes, the heuristics included disable the confidence mechanism temporarily to avoid it changing the weights of other notes so that a jump will not occur. The trills were added at random.



score performed	c b a g♯ a b c d c a b g a b c a g♯ a c a b g a f♯ g♯ f e d c b c a b a b g♯ a
expected accompaniment	e d c e a c f b e c d b c b a c e a e c d b c a b a b g♯ a - d - e - e - a
system response	e d c e a - f b e c d b c b a c e a e c d b c a b a b g♯ a - d - e - e - a

I

Fig. 2. Excerpt from the Prelude from *English Suite in A minor* (BWV 807) by J. S. Bach and the results from experiment 3. In the first row of the table, the trill marks show the random trills added to the piece by the performer. The difference between the expected accompaniment and system response is highlighted

4 Discussion and Further Work

The above results demonstrate how the system copes with the various situations that can occur during piano practicing, such as repeats, wrong notes and interpretation variations in performing the ornaments. When there are no mistakes in the performance, the system performs accurately. Where trills and ornaments were encountered the system was also able to provide the right accompaniment. In the case of repetitions of the same passage played with a number of mistakes, the system needed one or two successive notes to catch up with the performer. This was because the system needed to resolve the ambiguity caused by the new jump location. This is also what would be expected to be the case with a human accompanist.

In this paper we presented a piano score following system which is designed for piano students practicing on a MIDI instrument. However, instruments with MIDI interfaces are not very common and the existing ones are generally far more expensive than the more traditional instruments. In order to create a system that can be used with a wider array of instruments, a pitch detector can be used.

The polyphonic version of the described score following system is currently under development. One of the important advantages over the monophonic version will be the inclusion of rhythmical information to the score following task. This will allow students to practise a wide range of piano repertory without any restrictions.

The present score following system, which only makes use of the pitch information from the performance, has so far produced promising results. With the added capabilities of polyphonic music support, using duration information for increased robustness and a pitch, the system will be a valuable learning tool for many music students.

References

1. Bloch, J.J., Dannenberg, R. (1985). Real-Time Computer Accompaniment of Keyboard Performances. Proceedings of the 1985 International Computer Music Conference, International Computer Music Association, pp. 279-289.
2. Bora, U., Tufan, S., Bilgen, S. (2000). A tool for comparison of piano performances. Journal of New Music Research, 29, No.1, pp.85-99
3. Cano, P., Loscos, A., Bonada, J. (1999). Score-Performance Matching Using HMMs. Proceedings of the 1999 International Computer Music Conference, International Computer Music Association, pp. 441-444.
4. Dannenberg, R. (1984). An Online Algorithm for Real-Time Accompaniment. Proceedings of the 1984 International Computer Music Conference, International Computer Music Association, pp. 193-198.
5. Dannenberg, R., Mukaino, H. (1988). New techniques for Enhanced Quality of Computer Accompaniment. Proceedings of the 1988 International Computer Music Conference, International Computer Music Association, pp 243-249.
6. Dannenberg, R. (1991). Recent Work in Real-Time Music Understanding by Computer. Music, Language, Speech and Brain, Wenner-Gren International Symposium Series, Sundberg, Nord, and Carlson, ed., Macmillan, 1991, pp.194-202.
7. Dannenberg, R. et al. (1993). Results from the Piano Tutor Project. Proceedings of the Fourth Biennial Arts and Technology Symposium, (March 1993), pp. 143-150
8. Loy, G., Abbott, C. (1985). Programming Languages for Computer Music Synthesis, Performance, and Composition. Computing Surveys, Vol. 17, No. 2, June 1985
9. Orio, N., Lemouton, S., Schwarz, D. (2003). Score Following: State of the Art and New Developments. Proceedings of the Conference of New Interfaces for Musical Expression, NIME, Montreal, 2003, pp. 36-41.
10. Puckette, M., Lippe, C. (1992). Score Following in Practice. Proceedings of the 1992 International Computer Music Conference, International Computer Music Association, pp.182-185.
11. Puckette, M. (1995). Score Following Using the Sung Voice. Proceedings of the 1995 International Computer Music Conference, International Computer Music Association, pp. 175-178.
12. Raphael, C. (1999). Automatic Segmentation of Acoustic Musical Signals Using Hidden Markov Models. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.21, No. 4, pp. 360-370.
13. Raphael, C (2001). A Probabilistic Expert system for Automatic Musical Accompaniment. Journal of Computer and Graphic Stats. vol.10 no.3, pp.487-512
14. Roads, C. (1985). Research in Music and Artificial Intelligence. Computing Surveys, Vol. 17, No. 2, June 1985
15. Vercoe, B. (1984). The Synthetic Performer in the Context of Live Performance. Proceedings of the 1984 International Computed Music Conference, International Computer Music Association, pp.199-200.
16. AMuseTec website: http://www.musebook.com

Aspects of the Topology of Interactions on Loop Dynamics in One and Two Dimensions

Georg Essl

Media Lab Europe
Sugar House Lane
Dublin 8, Ireland
georg@mle.media.mit.edu

Abstract. This paper discusses aspects of topology as relevant for loop dynamics as they occur in physical modeling synthesis algorithms. Boundary and interaction point behavior is treated purely from a topological perspective for some dynamical systems in one and two dimensions.

1 Introduction

The purpose of this paper is to discuss aspects of topological ideas to interaction models of loop dynamics.

The theoretical approach is related but generally somewhat different to waveguide-like arguments (as reference see for example [1]). Specifically, we emphasize the topological structure of the problem over the details of the dynamics. This way arguments are valid for any dynamical situation for which the given topological structure holds. In this case the structure is one of closed and directional loops or orbits. Hence the only assumption made about the dynamics is that disturbances propagate, in some otherwise unspecified fashion, along such trajectories, either directly or in some useful approximation. This idea is in fact not new, but rather has been developed since Poincaré as a core aspect of the contemporary theory of dynamical systems. The goal here is to study simple one and two-dimensional situations that dynamically are very well-behaved yet occur frequently in physical modeling of musical instruments. That is, the periodic trajectories will be assumed to be structurally stable under perturbation and hence we are only concerned with aspects of the dynamics that is regular and integrable.

First I will develop the one-dimensional situation and introduce projection, lifting and covering arguments with respect to point interactions. Based on this discussion I will try to illustrate the benefit of postponing spectral arguments as essential features of interaction dynamics can be shown easily using the presented arguments. In addition the arguments are not confined to a specific dynamic situation and hence have a more general application that when a dynamical operator and domain shapes are starting points of the discussion.

Then I will extend this treatment to two-dimensional dynamics for which the dynamics can be represented to fall into families of tori in some related parameter-space. The action of point excitation will be extended to this situation.

A note on notation and treatment. I will use conventional notation and nomenclature as customary in the topology literature as far as possible, though in general, discussion and illustration of the core ideas by pictures will take precedence over detailed notation. The notation used is informed by texts by Hatcher [2, chap. 1] and Jänich [3, chap 9].

2 Remarks on the Relationship to Exact Wave-Equations

The relationship of orbit dynamics to the exact wave-equation is a very interesting and important one. However, it is also one that is difficult and in general yet unsolved. I will make no pretense that any contribution is made here to narrow this gap. The topological ideas are straight-forward for a class of billiard problems. For short wave-lengths the many properties of the wave-equation on the domain can be studied well using the billiard model. See for example [4, Section 2.7]. This was known to Keller and Rubinow already as they set up billiard style path constructions. The general case and specifically the behavior at long wave-lengths is far less understood but it is interesting that the discrepancy reported by Keller and Rubinow and repeated by Brack and Bhaduri [5] is small (less than 3%) and the error vanishes quite quickly [6]. In case of banded waveguides this error in the spectrum is usually avoided by tuning the model to exact frequencies which in turn corresponds to accepting spatial discrepancies [7].

3 Topology of Loop Dynamics in One Dimension

First I want to discuss the case of one dimensions. With dimensions, unless otherwise noted, I will mean the spatial dimensions of a related dynamical domain from which the topological construction is derived.

Let's start of with a one-dimensional domain of a line interval I. One could write $I \subset \mathbb{R}$ but strictly speaking we are not interested in the the metric properties that can be interpreted in $\mathbb{R}$, rather we are only interested in its connectivity. This also means that the line doesn't necessarily need to be straight. On the line I we will assume a waveguide-like dynamic of left and right-traveling disturbances, which travel along the line I staying on it through reflections at the domain limits ∂I.

By imposing a Euclidean distance metric and a constant wave speed on I with a suitable discrete spatial representation one would then immediately recover a standard loss-less waveguide model. Instead we want to study the properties in the absence of defined metric (length of string), wave speed (string tension and density). The advantage we gain are insights that hold for other situations as well, which share the same topology though potentially vastly different metrics, geometric layout and wave dynamics.

In the above description, the dynamics is uneventful except for the boundary points at which additional treatment has to be imposed. This special treatment can be generalized by seeing the one-dimensional line as a projection π of a two-dimensional loop, for example the circle $\mathcal{S}^1$ into the line I. This projection can be visualized as a circle lying flat and its two branches overlapping in ones view. We will call the operation of finding a smooth connected path in a higher-dimensional space *lifting*. The circle as a closed path is everywhere smooth, though we will keep track of the former position of the reflection by keeping singular markers at reflection points. While in the I an orbit apparently changes direction at the boundary ∂I this is not true on the lifted path S^1, where an orbit never changes direction. If the reflection points do nothing but keep disturbance within I, this constitutes already the first purely topological representation of the dynamics, meaning that all dynamical changes have been converted into smooth connectivity properties. We will call the cover *trivial* if reflections do not add dynamical behavior to a lifted space. An example of a waveguide model of this type would be the Karplus-Strong model without loss-filter.

In order to study interaction behavior on the lifted topology, we will first introduce another scenario. Instead of assuming a simple lifted space, we will assume that the lift consists of two layers which we will call *sheets*. The number of sheets are sometimes called the *degree* of a cover and if disconnected are labeled by sets of integers for each path-connected component [8, p. 464]. To illustrate a situation that leads to non-trivial covers and hence more than one sheet, consider the case of a loss-less string or a loss-less waveguide structure with Dirichlet boundary conditions. In this case, disturbances reflect with a sign inversion.

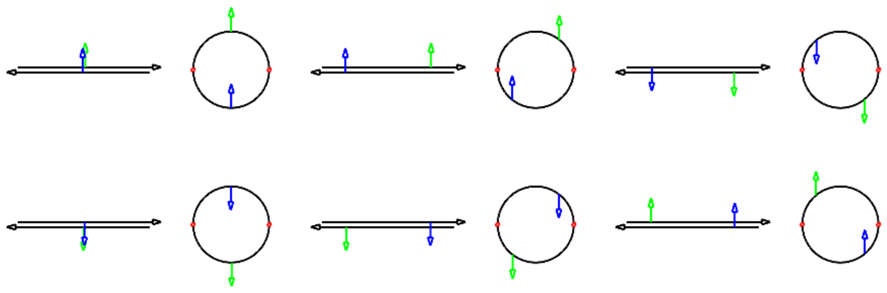


Fig. 1. Sequence of disturbance states on the line domain and its lifted circle domain.

Now let us place a disturbance of positive sign into the solution traveling in one direction and the same into the opposite direction. We trace the sign of the disturbance over time. At boundaries the impulses invert. We observe that the impulse traveling in one direction will always return with its sign inversed. Hence the two positive impulses trace two separate covers that are distinct and the total behavior can be seen as a projection of these distinct states onto the same covering space. See figure 2.

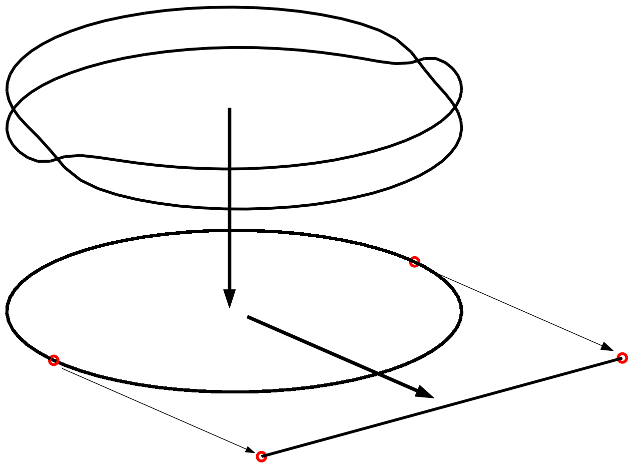


Fig. 2. The topology of two inverting reflections on a string.

To compare this with other boundary condition situations, we take Neumann conditions (no sign change at the boundary). Observe that we still have a non-trivial cover as positive and negative disturbance never share the same space. The cover maintains this separation everywhere as is depicted in Figure 3.

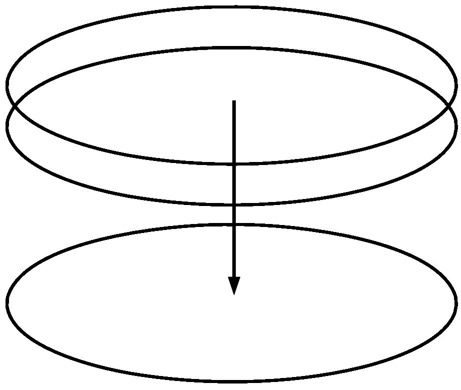


Fig. 3. The topology of two non-inverting reflections on a string or of a radial vibration of a cylinder.

This case is interesting because it can be used to visually prove the uniform velocity of a velocity excited string. The initial velocity distribution of some overall sign will be preserved in the lifted spaces and will be integrated over once per traversal through the covering loop, hence string will travel in the direction of the initial velocity distribution sign with the velocity defined by the traversal duration [9].

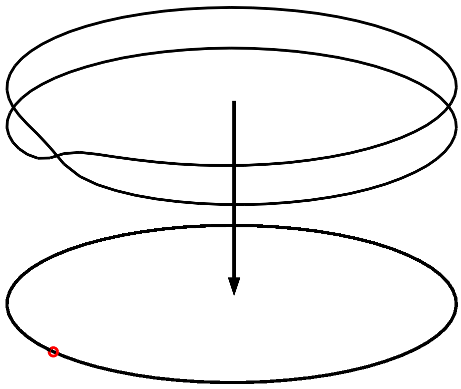


Fig. 4. The topology of one inverting reflections on a string.

A particularly interesting case is given by one Dirichlet boundary on one end combined with a Neumann condition on the other. In this case a disturbance will traverse the covering space twice to return to its original state. Hence, as opposed to the previous two cases the lifted space is thoroughly path-connected. See Figure 4. This implies the well-known fact that given the same metric and propagation behavior, this configuration has half the fundamental frequency. However we immediately also see that there are no two distinct sets of excitation configurations as opposed to the previous case. As a note (see [3, p. 57]), this topology is just the Möbius band which in turn can be described by the interval $I = [-1, 1]$ and the ends of the interval identified with a sign inversion $\alpha(i) = -i$, $i \in [-1, 1]$.

The degree of the cover the matched boundary conditions of either Neumann or Dirichlet type is the same, namely $\{1, 1\}$. The degree of the cover with mixed boundary conditions is $\{2\}$.

This already gives the topological result for the open pipe compared to the closed one. A disturbance needs to travel the loop twice to return to its origin in the same configuration and hence the wave-length is doubled for the pipe with one open end [10, p. 51].

Additionally the difference in degree between implies that all disturbance configurations can be reached by considering only one point in the lifted space of the mixed situation, whereas in the matched cases there are unreachable configurations. In practice one has however additional constraints on the excitations of the loops.

To describe the treatment we will assume that a notion of length is defined. This length describes how far disturbances travel over time. Additionally we will confine the current discussion to the situation that is familiar for the wave equation: Disturbances will travel to the left and right with equal contribution and with sign dependent on the particular variable chosen [1].

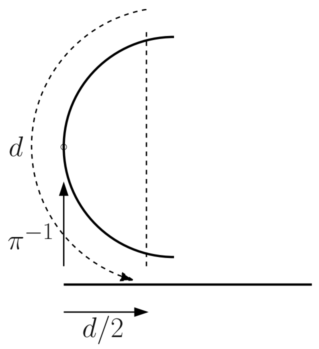


Fig. 5. Distance of coincidences to reflection points.

This is a situation where the projection of disturbance contribution in the lifted space *coincide*. Using this notion of distance we can then study when such coincidences will occur as disturbances propagate on the loops. Generally it is easy to see (for example in Figure 5) that disturbances coincide in the lifted space if they have the same distance from lifted domain boundaries. This situation can also be seen in Figure 1.

It also shows how an initial coincidence situation leads to another one after a half-rotation in the lifted space, reaching again a situation where the distance is symmetric and $d/2$ with respect to the second domain boundary. If the original interaction point was in the middle, i.e. $d/2$ both to the left and the right boundary. This is in fact a special situation. In Figure 6 we see the topology of the disturbance coincidence loop. It is a symmetric eight-shaped loop where the intersection point describes the lifted point of coincidence. In general the situation is somewhat more complicated.

The complete situation can be seen in Figure 7. The double-sheet eight with one double-intersection splits into a double-sheet of two path-connected loops that cross at two separate coincidence points. The crossing[1] has the form depicted in Figure 11 (a1). A single path-connected loop is depicted in Figure 8.

As is to be expected, the lifted loop does not self-intersect, as a single traveling disturbance in only one direction would never find an intersection point with itself. The loop is changing elevation at coincidence point and hence unfolds an apparent region of coincidence in the plane cover.

Coincidence arguments are of interest as they allow to make "spectral-like" arguments that are short-lived. At points of coincidence a disturbance can be annihilated by matching it with an equal but sign-inverted disturbance. The

[1] We note that this crossing has the form of the Vassiliev knot invariant [8], but knot-theoretic treatment will be work of the future.

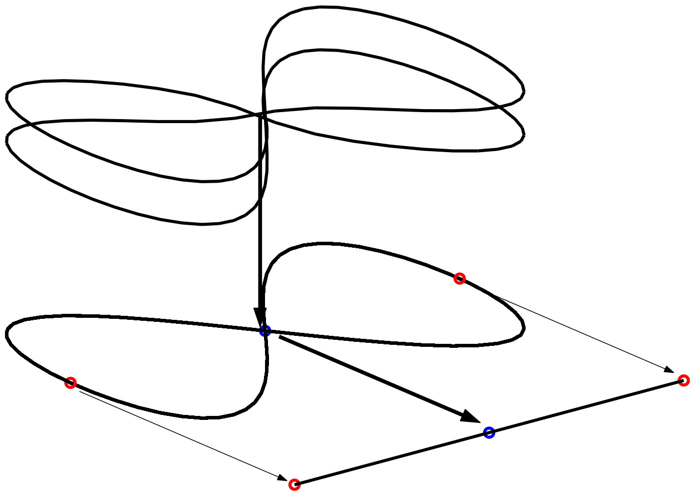


Fig. 6. Topology of disturbance coincidences at equal distance to both boundaries.

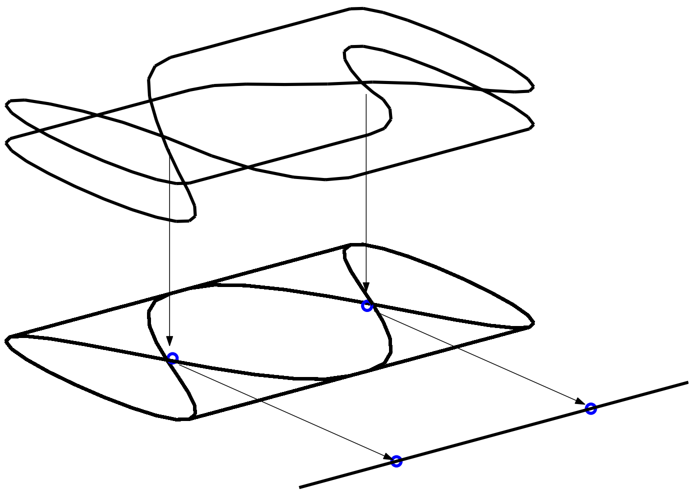


Fig. 7. Topology of disturbance coincidences at arbitrary distance to a boundary.

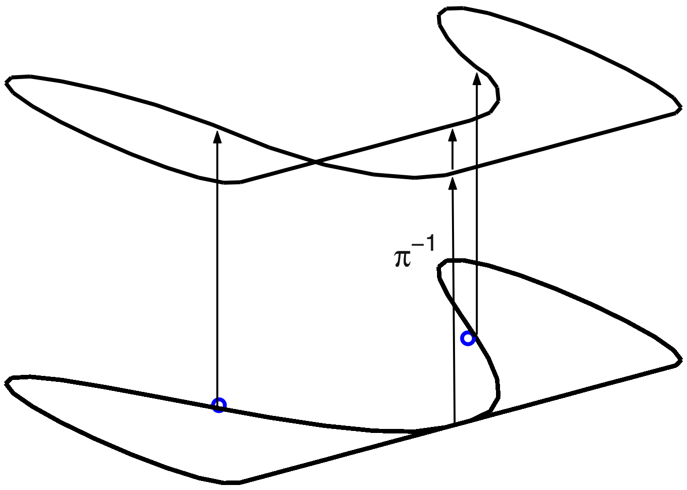


Fig. 8. One loop of the generic disturbance coincidence topology.

conditions of coincidence is defined by the topology described above plus the loop speed between coincidence points, which here is a global property. In the case of the center excitation such annihilation can happen every half-rotation at the same point, whereas otherwise it requires a full rotation for the configuration to return to the same spot. Similar arguments have been used to derive non-propagating excitations locally [11].

4 Topology of Loop Dynamics in Two Dimension

The situation naturally extends to two and more dimensions. Here we will only discuss this extension to two dimension because of its applicability to vibrating flat structures. It also already indicates how in general the extension behaves for higher dimensions.

In the one-dimensional case we lifted a line-domain into loops in two dimensions. In two dimensions we will principally be concerned with plane domains which will be lifted into loops in three dimensions. More precisely the concern is with a toroidal topology embedded in three dimensions. This is the flat, potentially layered 2-torus, meaning that the metric of the related dynamic is "flat" Euclidean space and that it is a 2-dimensional torus embedded in 3 dimensional space. The layers correspond to the sheets of circle covers discussed in the previous section.

The details how toroidal structures relate to dynamical situations has been known and has also been previously discussed in relation to physical models for sound synthesis using banded waveguides [12].

To quickly give an intuition of this relation, see Figure 9. A family of parallel rays will become straight through reflections if the plane is extended via mirror

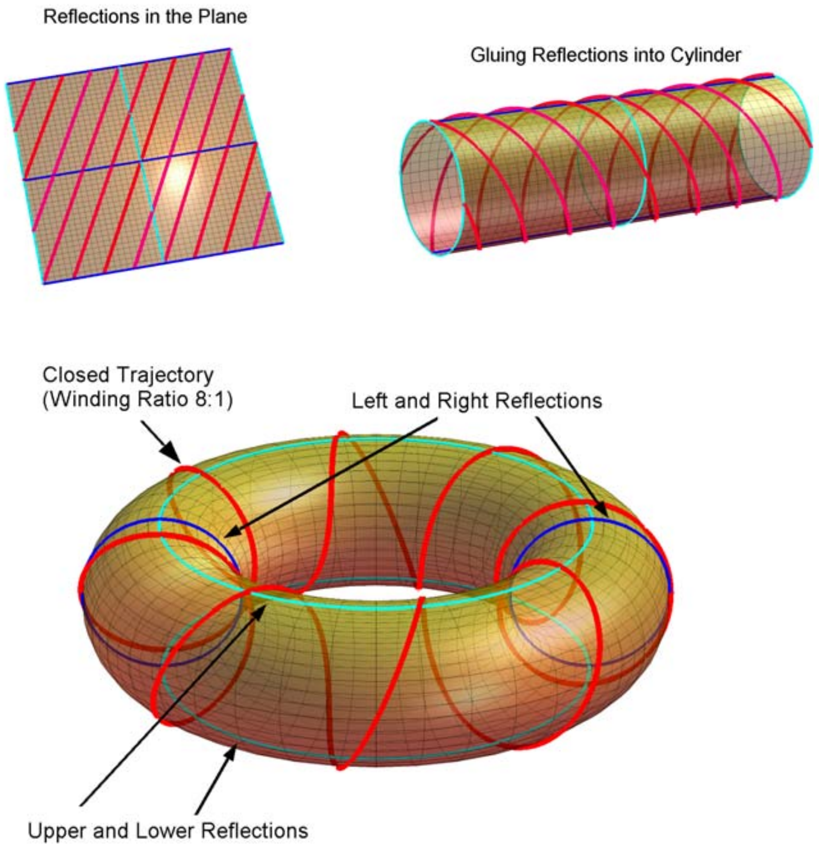

Reflections in the Plane
Gluing Reflections into Cylinder
Closed Trajectory
(Winding Ratio 8:1)
Left and Right Reflections
Upper and Lower Reflections

Fig. 9. Folding of path-connected families from a plane sheet into 2-torus by identifying edges representing reflections.

images. As the direction of the rays repeat after two reflections, these edges can be identified and form a tube, which we here depict to be have an extended volume, though the related dynamics is still flat. With the same argument the ends of the tube can be identified after two reflections and then form a torus. If the number of windings in both dimensions of the torus are integer, ray paths will close and form loops.

For our purpose here, we will not consider the varied connections of such loops to dynamical systems and refer to [6,12,7] for further details.

The torus has a cover of the square as follows from the construction above. When projecting the torus back to the cover to undo the construction we see that the four sides of the torus occupy the same space. In the case of mapping the loops onto the line we usually encountered two loop branches to coincide on the line. Reflections on the other hand are only concerned with one point on the loop, namely where the loop intersects with the reflection circle. Similarly in one dimensions the loop intersected with the two reflection points.

Hence we can extend the treatment of reflection points simply to the two-dimensional case by noting that a reflection constitutes a change in sheet of the torus. Neumann boundary in two dimensions corresponds to a single-sheet, unlayered torus. If the total number of crossings with reflection circles in both toroidal dimensions are even then we get two separate crossing sheets whereas if it is odd, we get a single path-connected 2-sheet loop, all in analogy with the one-dimensional situation.

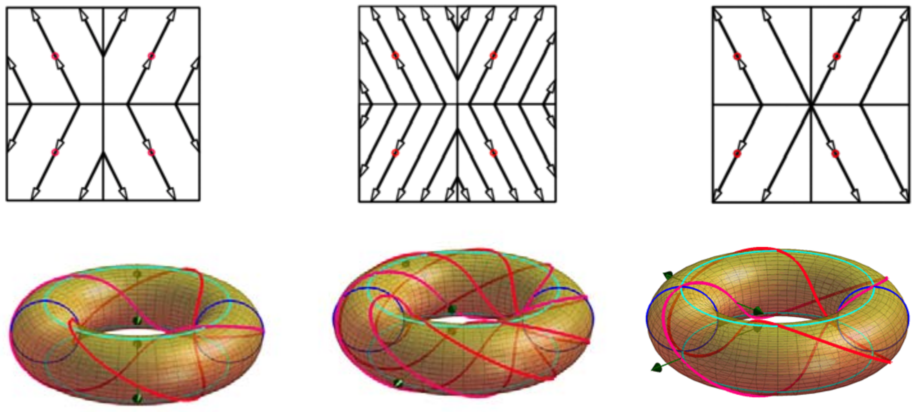


Fig. 10. Excitation point (center, left, far-left) and their loops. Top: cover. Bottom: torus.

Properties of placement of disturbances on the covering space can by seen in the depiction of Figure 10. In the case of center excitation and one off-center excitation we see that for the given winding numbers, the disturbances trace only two disjoint loops, whereas in the case of the other depicted off-center excitation one gets four disjoint loops. Clearly, these are the only two cases possible for integer winding numbers. On close inspection we see that in fact the center case and the off-center case with two disjoint loops differ in the way the excitation paths coincide. In the center case the disturbance contributions starting on the outside of the torus share the same loops, whereas in the off-center case the front outside shares with the back inside and the front inside disturbance contribution shares with the back outside. The respective conditions are captured in the following equations (1) and (2) with n being the overall sum of winding numbers of the loop on the torus, $d_{1,2}$ is the distance across reflection and $n_{1,2}$ are winding counts along independent torus dimensions:

$$d_1 = n_1, \qquad d_2 = n_2, \qquad n_1, n_2 \in \mathbb{N}, \qquad n_1 + n_2 < n \tag{1}$$

$$d_1 = n_1, \qquad d_2 = (2n_2 - 1)/2, \qquad n_1, n_2 \in \mathbb{N}, \qquad n_1 + n_2 < n \tag{2}$$

The coincidence loops are somewhat more complicated to depict than the ones in one dimensions. Like in the one-dimensional case we expect a crossing between paths from all possible directions to occur at a point of excitation. Given that four directions (up-left, up-right, down-left, down-right) are possible on the torus, this also defines the number of directed lines forming the crossing at the coincidence point as is depicted in Figure 11 (b1).

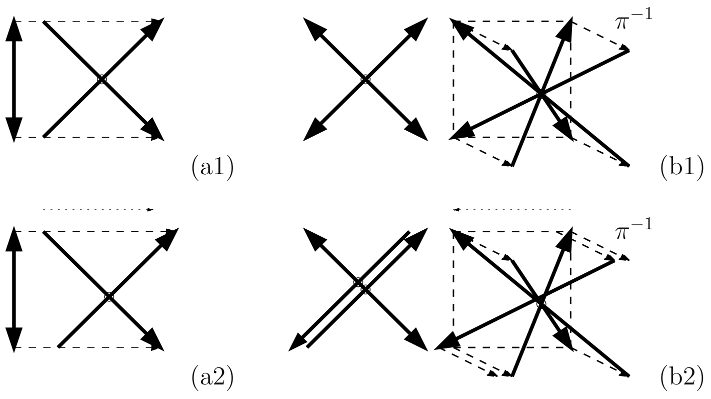


Fig. 11. The coincidence crossing in one and two dimensions. (a) shows the one dimensional two-path crossing and (b) shows the flat and lifted version of the four-path crossing of the two-dimensional case.

Depending on whether or not one of the conditions (1) and (2) hold, the coincidence point will intersect two or four otherwise disjoint loops away from it in both directions. The positions of coincidences cannot be as simply connected to reflections as in the one-dimensional case, but are easily observed on the torus topology. If we ascribe an overall loop-length, at least one such coincidence point will persist at the original location. Under what conditions more coincidence points exist is an open question. The length measures of all four path components need to match and coincide in the projected plane. Alternatively one can say that the knot doesn't persist under perturbation. This is illustrated in Figure 11 (b2) where one path is perturbed in the lifted space and then projected back. Notice that the coincidence does not persist. While in the one-dimensional case, equal length points from the reflection will always be forced by the projection into a point (see Figure 11 (a1-a2)), this is not the case in the plane. Hence it is a rather special condition as intersection points of trajectories constitute a sparse discrete set.

5 Conclusions

We discussed aspects of the topology of loop dynamics in one and two dimensions as they relate to effects of boundaries, excitation paths and disturbance coincidences of transportation type physical models for sound synthesis. Certain properties can be read immediately from such topologies without reference to the details of the dynamics involved.

Acknowledgments

Much thanks to Sile O'Modhrain for her support. This work was made possible by the kindness of employment of Media Lab Europe and access to its academic resources.

References

1. Smith, J.O.: Acoustic modeling using digital waveguides. In Roads, C., Pope, S.T., Piccialli, A., De Poli, G., eds.: Musical Signal Processing. Swets, Lisse, Netherlands (1997) 221–263
2. Hatcher, A.: Algebraic Topology. Cambridge University Press (2002)
3. Jänich, K.: Topologie. 7th edn. Springer, Berlin (2001)
4. Tabachnikov, S.: Billiards. Soc. Math. de France, Paris (1995)
5. Brack, M., Bhaduri, R.K.: Semiclassical Physics. Volume 96 of Frontiers in Physics. Addison-Wesley Publishing (1997)
6. Keller, J.B., Rubinow, S.I.: Asymptotic Solution of Eigenvalue Problems. Annals of Physics **9** (1960) 24–75
7. Essl, G., Serafin, S., Cook, P.R., Smith, J.O.: Theory of Banded Waveguides. Computer Music Journal **28** (2004) 37–50
8. Bar-Natan, D.: On the Vassiliev Knot Invariants. Topology **34** (1995) 423–472
9. Essl, G.: Velocity Excitations and Impulse Responses of Strings – Aspects of Continuous and Discrete Models. ArXiv preprint physics/0401065, http://arxiv.org/abs/physics/0401065 (2004)
10. Rayleigh, J.W.S.: The Theory of Sound. Volume II. Dover, New York (1945)
11. Essl, G.: Trapping and Steering on Lattice Strings: Virtual Slow Waves, Directional and Non-propagating Excitations. Physical Review E **69** (2004) 066601–1–6
12. Essl, G.: Physical Wave Propagation Modeling for Real-Time Synthesis of Natural Sounds. PhD thesis, Princeton University, Princeton, NJ (2002) Available also as Princeton University Computer Science Technical Report TR-659-02 at http://ncstrl.cs.princeton.edu/expand.php?id=TR-659-02.

Perceptive and Cognitive Evaluation of a Piano Synthesis Model

Julien Bensa[1], Danièle Dubois[1], Richard Kronland-Martinet[2], and Sølvi Ystad[2]

[1] Laboratoire d'Acoustique Musicale, Université Pierre et Marie Curie,
11 rue de Lourmel, Paris, France
{bensa, dubois}@lam.jussieu.fr
[2] Laboratoire de Mécanique et d'Acoustique, équipe S2M,
31 ch. Joseph Aiguier, Marseille, France
{kronland, ystad}@lma.cnrs-mrs.fr

Abstract. The aim of this work is to use subjective evaluations of sounds produced by a piano synthesis model to determine the perceptual influence on phenomena involved in sound production. The specificity of musical sounds is that they are intended for perception and judgments by human beings. It is therefore necessary, in order to evaluate the acoustic qualities of a musical instrument or a sound model, to introduce a research approach which takes into account the evaluation of the sound quality by human beings. As a first approach we synthesize a number of piano sounds. We then evaluate the quality of the perceived acoustic signal by questioning a group of persons. We hereby try to link the model's parameters to semantic descriptors obtained from these persons and to more classical perceptual signal descriptors. This approach should give a better understanding of how the model's parameters are linked to cognitive representations and more generally give new clues to cognitive descriptions of timbre of musical sounds.

1 Introduction

The piano is a complex instrument with a large number of mechanical elements, the majority of which contribute to the sound production. The physical characteristics of these elements together with their interaction influence the timbre of the piano sound. Thus, in order to give a precise description of the behavior of this instrument and effectuate a satisfactory sound synthesis, the totality of the physical phenomena that are part of the sound production ideally should be taken into account. However, due to the complexity of the sound production system, this is not possible. We therefore propose, thanks to sound modelling and synthesis, to determine the most important perceptual phenomena related to the sound production system and to evaluate the importance of each of them. This approach hopefully will give a better understanding of the relation between the physical behavior of the instrument and the perception of its sound quality, and hereby give clues to how the piano model can be simplified without loss of quality. This is crucial for the conception of high-quality synthesis models that are to be run in real-time.

As a contrast to non-intentional noises (or sounds) generated by different sources and generally considered as annoying, musical sounds are produced to be perceived and appreciated by human beings. Thus, in order to evaluate the sound quality, it is necessary to look for “subjective judgments” given by a number of subjects (professional musicians, amateurs or instrument makers) that listen to and compare the sounds. Although classical psychophysical approaches have been used for a long time to elaborate subjective evaluation methods, the analytic and parametric character of the sound samples that generally are implied in the tasks that the subjects are asked to perform, does not seem appropriate to the musical sounds that are studied here. The subjective evaluation used in psychoacoustics is generally based on stimuli which are analytically described within a multidimensional space given by physics. Such an approach mainly investigates low-level processing such as perceptual thresholds rather than high-level processing of complex sounds such as musical samples that cannot be reduced to a set of values identified by two or three physical dimensions. Although each stimulus is given by a unique physical description determined by the parameters of a physical model, there is a multitude of heterogeneous and poorly known principles of the organization of global perceptual judgments. In particular, the perception of timbre of a musical instrument is not only related to sensations linked to the characteristics of the physical signal, but also to criteria associated to interpretation processes and to knowledge achieved in a particular cultural community. We have therefore deliberately chosen to take these “high quality” characteristics into account when modelling the acoustic qualities of piano sounds. This means that global judgments are considered, referring not only to what is perceived in an analytical listening condition, but also to the sensation that the subjects approve as a result of their personal experience, their knowledge and their expertise.

To obtain this subjective evaluation, we used a methodology that has already been validated on visual and olfactory stimuli [1,2]. This methodology relies on theoretical assumptions regarding cognitive categories and their relations to language [3]. The psycholinguistic analysis of verbal comments that subjects produce as answers to acoustic stimulations can be considered as an access to cognitive representations. We therefore processed a free categorization task: subjects were asked to freely classify the stimuli according to personal criteria, and to comment their final classification.

The free categorization method has a theoretical frame which is adapted to the rather unknown character of cognitive structures of musical objects [4,5,6]. As a contrast to estimation methods where subjects are asked to adapt their judgment to a pre-defined scale, the tasks of our method make it possible to induce properties of pertinent physical stimuli (not necessarily known) of the cognitive representations. In fact, the subjects are given the freedom to choose their own subjective measure to describe the stimuli. Thus, we are making the hypothesis that for the same class of rather complex sounds (for instance different sounds from the same instrument), subtle timbre variations can be identified from a semantic analysis of the descriptions given by the subjects although they are difficult to distinguish by more classical descriptors (obtained from the conceptualization of physical science). In fact, as Jensen mentions [7], there is a need to study subtle timbre variations within the same instrument since several timbre parameters often are more similar for sounds from different instruments with the same pitch than for sounds from the same instrument with a different pitch. Hence, the comments from the subjective evaluations will make it possible to discover different domains of knowledge involved in the evaluation of sound quality as a function of the subjects' different expertise.

In the first part of the paper we shortly describe the main physical phenomena involved in the piano sound production and the model we use to produce the set of stimuli. We here focus and confront two features of piano sounds: inharmonicity and "phantom" partials. We then show how classical timbre descriptors (centroid and spectral flux) may be used to characterize each stimulus. Finally, we describe the categorization task and give some preliminary results leading to a discussion on how the different cognitive categories are related to the model parameters.

2 The Synthesis Model and the Control Parameters

The piano consists of a large number of components, each one having a role that is more or less important to the sound production mechanism. The piano is a struck string instrument. Each note can use either one, two, or three strings with physical properties that differ from one note to another. The resulting timbre depends namely on the interaction between the hammer and the string, the coupling between the strings, and the way in which the sound radiates from the soundboard. Many studies (see i.e. [8,9]) have described the behavior of the varied elements of the piano and we here refer to those publications for a more precise descriptions of the acoustics of this instrument. Different types of sound synthesis models of the piano simulating phenomena involved in sound production have been proposed [9,10,11,12,13]. Signal models are generally computationally efficient enough to run in real time and can be very accurate in reproducing the sound of an existing piano. However, these types of models fall short when it comes to incorporating the player into the control-instrument loop. Since they make no direct link between the player's actions and the physics of the instrument, important playing conditions have no effect on the produced sound. Physical models on the other hand, have the advantage of simulating the interaction between player and instrument, although this comes at a computational cost (though this cost is becoming less of an issue as computers become increasingly powerful). The parameters of physical models are difficult to accurately estimate from measured signals, and their sound quality often is poorer than for signal models. The quality of the synthesis depends on how accurately the acoustic system is taken into account in the physical model. Though algorithms are becoming more efficient and computer computation ever more powerful, the balance between sound quality and algorithmic complexity is still delicate. Thus, one of the first motivation of this study is to obtain a classification of the different phenomena involved in piano sound production. This classification is of great importance for the perceptual quality of real-time physical models.

Here, we would like to confront the perceptual effect of two phenomena involved in sound production: the string stiffness and the tension modulation, which respectively lead to the inharmonicity of the piano spectrum and the so-called "phantom" partials. Inharmonicity is a well-known characteristics of piano spectra. The "harmonics" of the piano sound are not exact integral multiples of the fundamental frequency. The whole spectrum is stretched and its components are called partials. This inharmonicity contributes to the piano tone and is mainly responsible for its specificity [16]. The stretched tuning of piano strings can be almost entirely attributed to their inherent stiffness [17], which leads to a dispersion of waves during the propagation. For small stiffness, the modal frequencies of the string are [16]:

$$f_n = nf_0\sqrt{1 + Bn^2} \tag{1}$$

where f_n is the modal frequency, f_0 the fundamental frequency, n the partial index and B the inharmonicity factor.

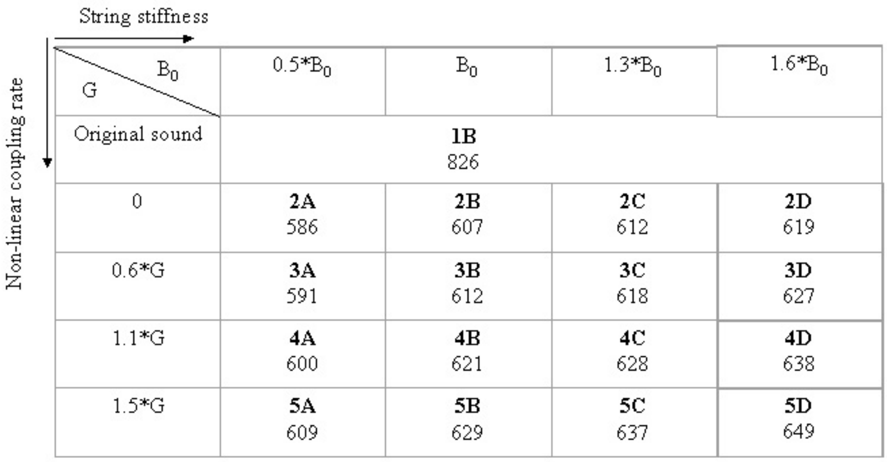
The models commonly used to simulate piano string vibrations take into account the propagation of transverse waves (including stiffness and losses), leading to a spectrum of inharmonic partials. However, a close inspection of Fourier spectra of recorded piano sounds shows that a number of partials cannot be related to the transverse modes of the string and are not foreseen by the linear theory. Moreover, those partials seem to contribute to the piano timbre, especially for low-pitched notes. Studies dealing with this phenomenon make the assumption that the appearance of those partials, also called "phantom" partials [18] (we will use this terminology in this article, even if this is maybe not the most appropriate term) are somewhat related to tension variation in the string. Due to transverse waves, the shape of the string changes during the motion, and the tension is modulated. This modulation introduces a coupling between the transverse and longitudinal modes of the string, giving rise to new partials [19,20]. Both of the phenomena described previously affect the frequency, amplitude and damping of the spectral components of a piano sound. We here investigate how they contribute to the perception of the piano timbre and how their combination affects the perceptual judgement of the listeners.

The model we use is based on analysis and synthesis techniques described in [13]. For this test, we worked with only one piano note (B1), recorded in an anechoic room. This note corresponds to two strings tuned to a very close pitch and thus, the recorded sound exhibits beating phenomena. Moreover, we localized many "phantom" partials on the corresponding spectrum. We accurately analyze the original sound using technique based on time-frequency representation and parametric methods given in [13,21]. As a first step, we estimate the modal parameters of each partial of the inharmonic spectrum, i.e. two frequencies, damping coefficients and initial amplitudes. In a second step, we localize the "phantom" partials (using an automatic algorithm we have developed) and estimate the corresponding modal parameters. The synthesis model is made of two different parts: a digital waveguide model [14] modelling the propagation of transverse waves in the string and an additive signal model allowing to introduce "phantom" partials [15]. This kind of signal model exhibits three main advantages for the needs of the listening test we would like to carry out. First, the synthesized sound is perceptually extremely close to the original sound: it is actually important that the subjects could not identify those synthesized sounds as "synthetic" stimuli. Secondly, the signal model offers the possibility of modifying independently the contribution of each phenomena and thus allows to regularly increasing different parameter values. Third it is possible, at least for partials coming from transverse waves, to modify their frequencies with respect to physical laws by taking into account inharmonicity law [16] and attenuation law (the physical descriptions of the "phantom" partials found in the literature are not accurate enough to allow a physical control of their modal parameters).

Using this model, we have synthesized sounds for different inharmonicity factor and different level of "phantom" partials. B is the inharmonicity factor (with $B_0 = 2.4176.10^{-4}$ its value for the original sound) and G is the global gain (G_0 is the value of the original sound) mixing "phantom" partials with "regular" partials. We obtained 17 stimuli as shown on Fig. 1 (labelled 1B, 2A, 2B...). The variation range of the two parameters has been chosen to cover a wide range of perceptual effects, from a sound with very weak inharmonicity and no "phantom" partials (2A) to a sound with exaggerated inharmonicity and a high level of "phantom" partials (5D). 4B is the closest sound to the original sound 1B from the model parameters point of view.

3 Timbre Description Using "Classical" Descriptors

Timbre is probably one of the the most well-known and least understood attribute of the quality of a sound. It refers to those aspects of a sound other than pitch, loudness, perceived duration, spatial location and reverberant environment [22] and can be regarded as the feature of an auditory stimulus that allows us to distinguish one source from another when the other five perceptual features are held constant. Thus timbre can be considered as the "tonal color and texture" [23] that allows us to distinguish two different instruments playing the same note. Although a lot of work has been done to describe timbre [24,25,26], it is not yet a fully understood component of auditory perception. One of the reasons for this is probably the fact that the cognitive processes of timbre classifications depend on the experience of each listener [3]. This is one of the motivations behind our attempt to link timbre to semantic descriptors by means of a piano synthesis model in order to approach a cognitive description of timbre of musical sounds. This is particularly important when studying subtle timbre variations within the same instrument since several timbre parameters often are more similar for sounds from different instruments with the same pitch than for sounds from the same instrument with a different pitch [7]. As a starting point to this investigation we will study some parameters that are traditionally used to describe timbre. Some of the most significant parameters commonly mentioned


String stiffness
Non-linear coupling rate

G \ B_0	0.5*B_0	B_0	1.3*B_0	1.6*B_0
Original sound		**1B** 826		
0	**2A** 586	**2B** 607	**2C** 612	**2D** 619
0.6*G	**3A** 591	**3B** 612	**3C** 618	**3D** 627
1.1*G	**4A** 600	**4B** 621	**4C** 628	**4D** 638
1.5*G	**5A** 609	**5B** 629	**5C** 637	**5D** 649

Fig. 1. The stimuli as a function of the inharmonicity factor B (with $B_0 = 2.4176.10^{-4}$ the original value) and the global gain G of the spectral "phantom" components ($G = 0$ means no "phantom" components, $G = G_0$ means the same level of "phantom" components as for the original sound). Spectral centroid in Hz for the sixteen synthesized sounds and the original sound.

in the literature are the spectral centroid, the attack time, the spectral flux and the spectral irregularity [25]. We have chosen to calculate two of these timbre descriptors from the 17 synthesized piano sounds that have been made for this particular study, namely the spectral centroid and the spectral flux. Since the attack time of the synthesized sounds used in this study is the same for all the piano tones, and since the irregularity of the spectrum (variations between the amplitude of the spectral components) is constant for the 17 sounds, we have here been considering the spectral centroid (SC) and the spectral flux (SF). The spectral centroid is defined as [24]

$$SC = \frac{\sum_k kA_k}{\sum_k A_k}, \tag{2}$$

(with k the partial index and Ak the spectral partial amplitude) and is often said to be related to the brightness of the sound. The spectral flux is a mean value of the variation of the spectral components as a function of time [25]. This means that it describes the attenuation of the spectral components which is of great importance for the perception of the sound. In this article we have chosen to calculate the spectral flux from successive centroids estimated at different times, meaning that this parameter can be considered as a time varying brightness. As mentioned in Sect. 2, the 17 different sounds are obtained for the same pitch, by varying the inharmonicity B and the phantom components represented by the gain factor G.

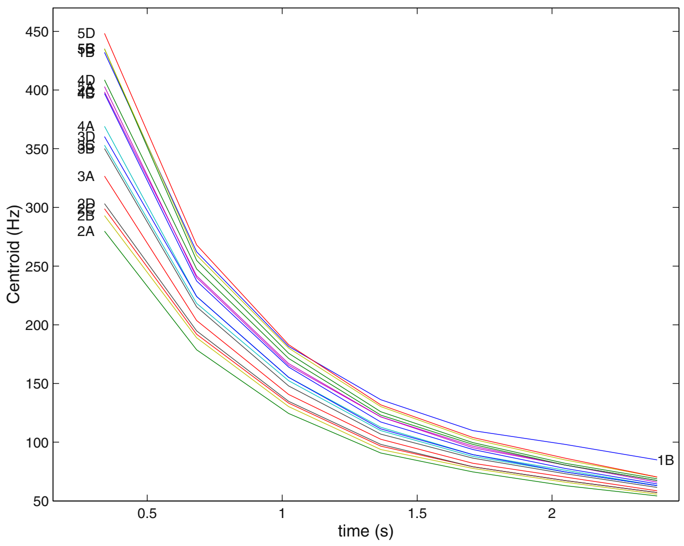
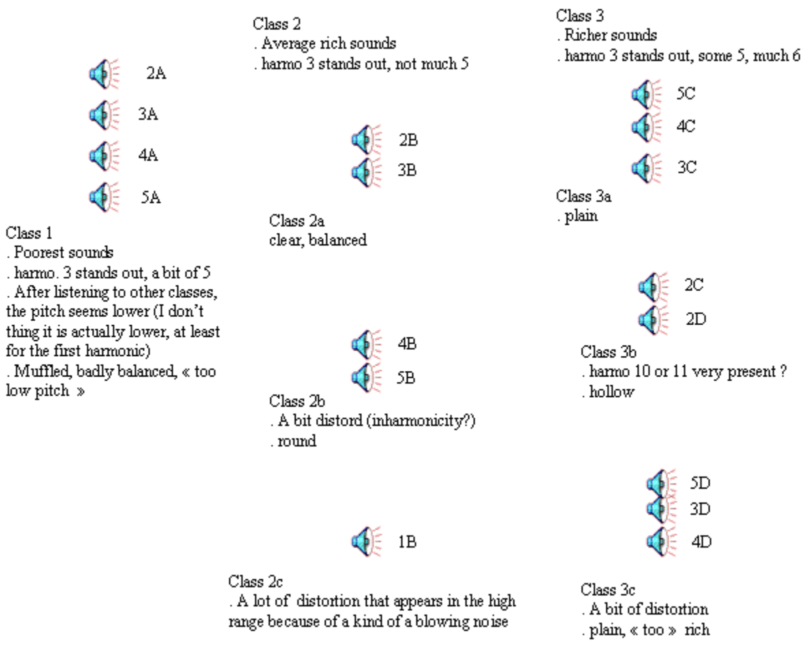


Fig. 2. Spectral flux represented by the spectral centroid as a function of time for the 17 different sounds.

Fig. 1 shows the values of the spectral centroid for the different synthesized sounds (mean value for 6 *ms* of the sound). As expected the centroid increases for increasing inharmonicity since the spectral components get more and more separated, and it also increases when the gain factor increases, since there are more and more "phantom components" for higher frequencies. We can also notice that stimuli 1B has a higher value of its spectral centroid. Fig. 2 shows the spectral flux represented by the spectral centroid as a function of time. One can observe that during the attack of the sounds (1st second), five different groups of sounds can be distinguished, namely 5D-5C-5B-1B,4D-5A-4C-4B, 4A-3D-3C-3B, 3A, and 2D-2C-2B-2A. These groups are rather coherent with the physical parameters since they more or less correspond to lines in the table (Fig. 1). This seems to indicate that the spectral flux varies with G_0 (gain of the phantom partials), but does not depend on the inharmonicity factor. After one second there is no specific group of sounds to observe and the spectral flux almost becomes the same for all of the synthesized sounds. The spectral flux of the original sound is higher than those of the synthesized sounds in the last part of the sound, but this doesn't seem to be perceptually important since listeners tend to pay more attention to the attack of the sound (as we will see in the next section).

4 Free Categorization Task: Protocol and First Results

The subjective evaluation was carried out on subjects using a free categorization task of the stimuli described above. Subjects were asked to freely sort the 17 piano sounds, symbolized by a schematic speaker on a screen (see Fig. 3) into groups and subgroups, and when the task was completed, to write down the reasons of their choice. When clicking on the speaker, the subjects could listen to the sound (through two "real" speakers) and interrupt it at their convenience by clicking again. The subjects could form as many categories as they wanted. One example of such a categorization is given in Fig. 3 below (results for subject SX, a pianist).



Class 1
. Poorest sounds
. harmo. 3 stands out, a bit of 5
. After listening to other classes, the pitch seems lower (I don't thing it is actually lower, at least for the first harmonic)
. Muffled, badly balanced, « too low pitch »

Class 2
. Average rich sounds
. harmo 3 stands out, not much 5

Class 2a
clear, balanced

Class 2b
. A bit distord (inharmonicity?)
. round

Class 2c
. A lot of distortion that appears in the high range because of a kind of a blowing noise

Class 3
. Richer sounds
. harmo 3 stands out, some 5, much 6

Class 3a
. plain

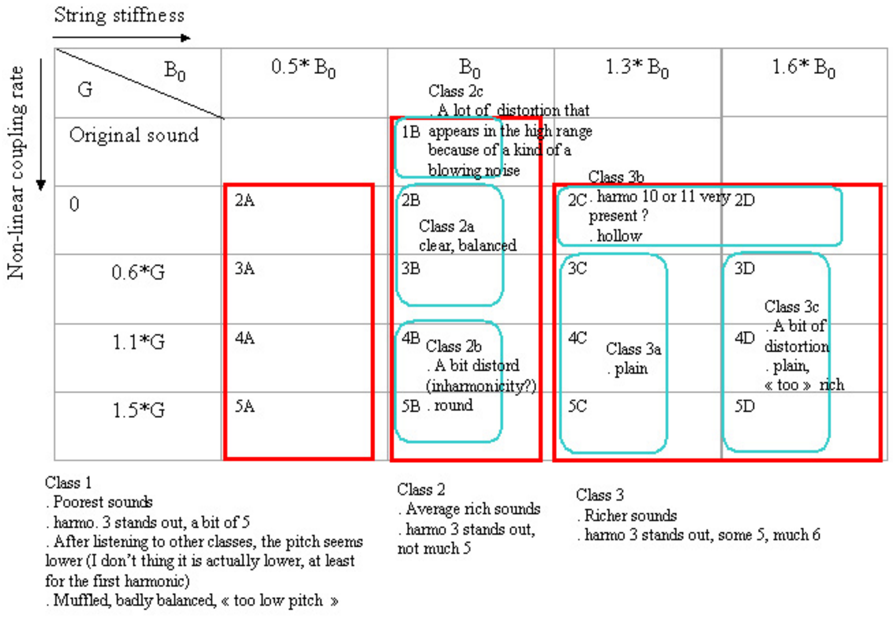
Class 3b
. harmo 10 or 11 very present ?
. hollow

Class 3c
. A bit of distortion
. plain, « too » rich

Fig. 3. Results of the free categorization task given by SX, a pianist.

The experiment is presently running and we here present the results obtained on 21 subjects, 7 pianists, 8 musicians non-pianists and 6 non-musicians. The full experiment will include at least 30 subjects, in order to contrast the different principles of categorization involved in the task and to elicit the characteristics that structure the cognitive specificities of these diverse populations. We therefore present the categories and comments of a pianist and a non-pianist (SX and SY as reported in Figs. 4 and 5) mapped on the matrix of controlled physical parameters involved in the construction of the experimental stimuli. As shown in Fig. 4, the first level of cognitive categorization for the pianist subject SX fits the physical criterion of inharmonicity, and the categorization that can be attributed to "phantom" partials is subordinate at a second level of categorization. However, such a categorical structure is not observed on the non-pianist subject

SY (Fig. 5) where, except for the lowest value of the inharmonicity factor, the 3 categories (2B, 2C), (2D, 5B, 5C and 5D), (3C, 3D, 4C and 4D) integrate the two physical dimensions with respect to their values on a single level. These two preliminary results favor the hypothesis that criteria of categorization highly depend on subjects' experience and therefore that different cognitive categories can emerge from the same set of stimuli univocally described in terms of physics.



String stiffness
Non-linear coupling rate
B_0
G
$0.5^* B_0$
B_0
$1.3^* B_0$
$1.6^* B_0$
Original sound
0
0.6*G
1.1*G
1.5*G
1B 2A 3A 4A 5A 2B 3B 4B 5B 2C 3C 4C 5C 2D 3D 4D 5D
Class 2c
. A lot of distortion that appears in the high range because of a kind of a blowing noise
Class 3b
. harmo 10 or 11 very present ?
. hollow
Class 2a
clear, balanced
Class 2b
. A bit distord (inharmonicity?)
. round
Class 3a
. plain
Class 3c
. A bit of distortion
. plain,
« too » rich
Class 1
. Poorest sounds
. harmo. 3 stands out, a bit of 5
. After listening to other classes, the pitch seems lower (I don't thing it is actually lower, at least for the first harmonic)
. Muffled, badly balanced, « too low pitch »
Class 2
. Average rich sounds
. harmo 3 stands out, not much 5
Class 3
. Richer sounds
. harmo 3 stands out, some 5, much 6

Fig. 4. SX (pianist) categorization projected on the representation of the sound samples as a function of the model parameters.

We can also already notice (as shown on Fig. 5) that the mapping of the cognitive categorization onto the physical description is not continue. 2D and 5D are grouped together within the same subcategory even if they are constructed on two extreme values of the "phantom" partial level. Considering the verbal comments, this "incoherence" means that subjects categorize the two stimuli together because they are "synthetic" or "bizarre", that is, they are different from what can be considered by the subjects as "common" sounds of a "good piano". In other words, cognitive categorization not only relies on "intrinsic" properties of the stimulus but also on similarities or discrepancies from "common" stimuli that the subjects frequently encountered and therefore experienced and memorized. These observations indicate that, for some values of a given physical parameter (here from 1,3*$B0$), the cognitive categorization is not related in a monotonous manner to a physical parameter, but operates according to mental representations elaborated from the subject's previous experiences. Moreover, the traditional timbre descriptors of the signal proposed in Sect.3 can not explain the categorization given by the subjects. We can thus find stimuli of very different centroids or spectral flux within the same category. The non-pianist subject sometimes seems to choose to group stimuli with respect to their inharmonicity (group 2A, 3A, 4A, 5A), sometimes with respect to the similarities of their centroids (group 3C, 3D 4C, 4D). The relation between groups given by the first seconds of the spectral flux and the subject categorization is not trivial. Thus, 1B and 4B have been grouped whereas their centroids as well as their spectral flux are different. These observations indicate that cognitive categorization can give us new ideas when looking for meaningful timbre descriptors to distinguish similar sounds. As already mentioned the cognitive categorization partly relies on similarities or discrepancies from "common" stimuli, meaning that one possible timbre descriptor could be related to a predefined distance from a reference sound.

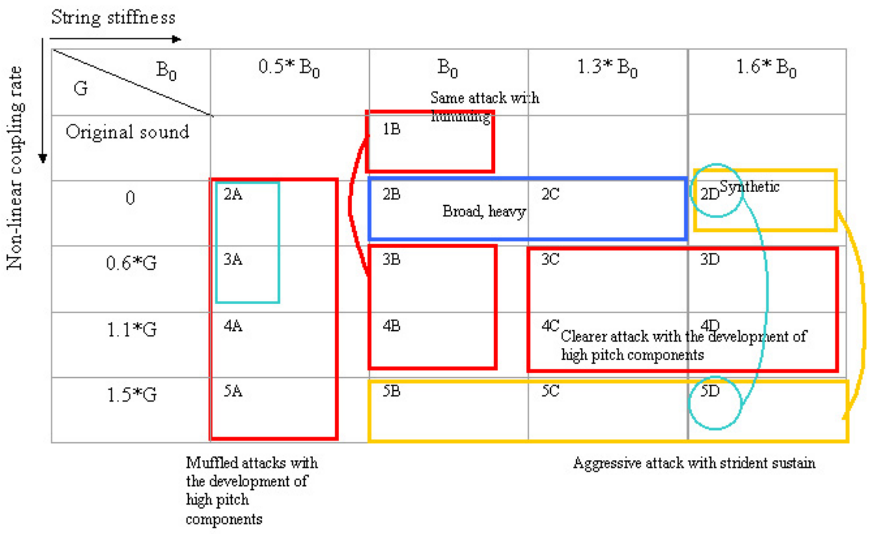


Fig. 5. SY (musician non-pianist) categorization projected on the representation of the sound samples as a function of the model parameters.

For the full experiment on a higher number of subjects, we will have to identify the semantic properties attributed to the same physical descriptions by subjects with different expertise and knowledge. In particular, we could like to verify the hypothesis stipulating that the categorization given by the experts (pianists) are more closely linked to the structures described by the physical parameters, and especially when it comes to the inharmonicity, while the non-experts used a different categorization strategy associating physical parameters of the model with a semantic interpretation integrating other criteria like the spectral centroid. The analysis of the verbal comments will be used to complete our interpretation.

As opposed to free audio categorization tests which have been done before, where subjects tend to classify stimuli regarding to the nature of the supposed source which produced them (door shock, engine noise [27]), stimuli are here in general grouped with respect to their perceptive properties ("muffled", "broad", "round"...). This is due to the fact that timbre variations, and consequently categorization levels, are here extremely subtle. The qualitative properties of stimuli coming from a same source (piano), are treated in an "intensionnal" way. Thus, the aim is here to listen to the sound itself as a contrast to an indicial way of listening [28]. Certain subjects sometimes can refer to a particular kind of source like "synthetic piano" as a qualification of a stimulus, meaning that they base the criteria of extensional classification of the sound producing system rather than on the sound itself.

If we look closer at the results of the free categorization task, for different values of our two "physical dimensions", we can state that

– when subjects judge the sound as a function of the inharmonicity, they use expressions like "muffled", "poor" and "mate" for the least inharmonic sounds, "between muffled and clear", "average rich sound" for the medium inharmonicity and "clear" or "distorded", "richer sound" for a maximum value of inharmonicity. An increase in inharmonicity therefore seems to "brighten" or "clear" the sounds from a perceptual point of view.
– when subjects judge the sound as a function of the "phantom" partials (which corresponds to the second categorization level for pianist subjects), the sounds with few partials are classified as "hollow", "damped", while for higher amounts of "phantom" partials they are classified as "clear", "round", "plain", "balanced" and finally for the maximum value of "phantom" partials as "slightly metallic", "aggressive". An increase in the number of "phantom" partials therefore seems to converge with an increase in the inharmonicity, constructing categories of more and more bright sounds.

The second criterion (presence of "phantom" partials) contributes to the construction of categories in which the tendency "muffled" or "dark" given by the weak levels of inharmonicity is corrected and renormalized. Finally, for the category constructed with a strong inharmonicity, the extreme stimuli are considered as "synthetic". Again these terms demonstrate the important gap between the norm of the human category jugdment.

5 Conclusion and Perspectives

This study shows the major difference between physical and cognitive descriptions, i.e. the dimensional character of the first one and the categorical character of the other. This difference can be seen by the deviations from the value which is considered as "normal" (sound close to the original one) within the same category of values in different registers. Thus, from a certain "level", the categorization induces a discontinuity compared to the physical parameters. From this separation in their categorical belonging, the stimuli are classified as a function of their distance to a prototype sound (either previously memorized from the experiences of the subjects, or constructed as a mean element within the group of stimuli). In other words, different categorical structures can correspond to the unique description of the stimuli in the physical space. These structures depend on different strategies that supposedly rely on the variations in expertise and experience of the different subjects. Thus, the non-pianist subjects construct their categories differently from one part to the other of the inharmonicity level. At this level the categorization is effectuated from the stimulus sharing the values of the two parameters of the model taken simultaneously. When the subjects possess a very high degree of expertise, the processes of categorization are no longer related to the qualitative properties of the signal, but on extensional criteria belonging to categories of objects with a close differentiation (here different piano types). It is therefore necessary to pay attention to the characterization of different groups of subjects. We have also noticed that the use of this diversity of categorization processes was sensitive to the characteristics of the stimuli. Thus, certain stimuli are for instance too far from the prototype, too atypical, have a construction that is too distant for the subjects to be able to constitute a representation of a "real" sound and correspond to an indication of a source. In such cases they are treated analytically, directly on the properties of the signal (already observed in visual experiments [3]). When a larger number of subjects are to be tested, it will therefore be necessary to take care of the validity of the representations of the different stimuli related to the "natural" sounds. We finally will strive to find new descriptors of the acoustic signal which represent categorical structures of the cognitive analysis, regrouping all of the parameter values that do not necessarily correspond to the same physical dimension. Finally, the coupling between the categorical analysis and the verbal comments will make it possible to extract semantic descriptors qualifying the perceptual effects of the two phenomena studied here. This will further give new clues to the definitions of new timbre descriptors adapted to subtle timbre variations of piano sounds.

References

1. Rosch, E.: Principles of categorization. In: Lloyd, B.B., Erlbaum , L. (eds.): Cognition and categorization. Hillsdale (1978) 27–47
2. Dubois, D. (ed.): Sémantique et Cognition. Editions du CNRS, Paris (1991)
3. Dubois, D. (ed.): Catégorisation et cognition : de la perception au discours. Kimé, Paris (1997)
4. Guyot, F.: Etude de la perception sonore en termes de reconnaissance et d'appréciation qualitative : une approche par la catégorisation. Thèse de doctorat, Université du Maine, Le Mans (1996)

5. Maffiolo, V.: De la caractérisation sémantique et acoustique de la qualité sonore de l'environnement sonore urbain. Thèse de doctorat, Université du Maine, Le Mans (1999)
6. Gaillard, P.: Etude de la perception des transitoires d'attaque des sons de steeldrums : particularités acoustiques, transformation par synthèse et catégorisation. Th'ese de doctorat, Université de Toulouse II, Toulouse (2000)
7. Jensen, K.: Timbre models of musical sounds. Ph.D. thesis, DIKU press, Copenhagen, Denmark (1999)
8. Askenfelt, A. (ed): Five Lectures on the Acoustics of the Piano. Royal Swedish Academy of Music, Stockholm (1990)
9. Chaigne, A., Askenfelt, A.: Numerical simulations of struck strings. I. A physical model for a struck string using finite difference methods. J. Acoust. Soc. Amer, Vol. 95(2). (1994) 1112–1118
10. Smith III, J. O., Van Duyne, S.A.: Commuted piano synthesis. Proc. Int. Computer Music Conf., Banff (1995)
11. Avanzini, F., Bank, B., Borin, G., De Poli, G., Rocchesso, D.: Musical instrument modeling: the case of the piano. Proc. of the worshop on current research directions in computer music. MOSART Research training network (2001)
12. Bank, B.: Physics-based sound synthesis of the piano. Master thesis, Budapest University of Technology and Economics, Hungary (2000). Published as Report 54 of HUT Laboratory of Acoustics and Audio Signal Processing. URL:http://www.mit.bme.hu/ bank
13. Bensa, J.: Analysis and synthesis of piano sounds using physical and signal models. Ph.D. thesis, Université de la méditerranée (2003). URL:http://www.lma.cnrs-mrs.fr/∼bensa
14. Smith, J. O.: Digital Waveguide Modeling of Musical Instruments. Available online at http://ccrma.stanford.edu/˜jos/waveguide
15. Bensa, J., Daudet L.: Efficient modeling of "phantom" partials in piano tones. Proc. of the International Symposium on Musical Acoustics, Nara, Japan (2004).
16. Fletcher, H., Blackham E.D., Stratton S.: Quality of Piano Tones. J. Acoust. Soc. Amer, Vol. 34(6). (1961) 749–761
17. Young, R. W: Inharmonicity of plain wire piano strings. J. Acoust. Soc. Amer, Vol. 21. (1952) 267–273
18. Conklin Jr., H.A.: Generation of partials due to non-linear mixing in stringed instrument. J. Acoust. Soc. Amer, Vol. 105(1). (1999) 536–545
19. Valette, C., Cuesta, C.: Mécanique de la corde vibrante. Traité des nouvelles technologies. Série Mécanique, ed. Hermès (1993)
20. Bank, B., Sujbert, L.: Modeling the longitudinal vibration of piano strings. Proc. Stockholm Music Acoustics Conf. (2003)
21. Aramaki, M., Bensa, J., Daudet, L., Guillemain, Ph., Kronland-Martinet, R.: Resynthesis of coupled piano strings vibrations based on physical modeling. J. of New Music Research, Vol. 30(3). Swets & Zeitlinger, (2001) 213–226
22. McAdams, S.: Recognition of sound sources and events. In: McAdams, S., Bigand, E. (eds.): Thinking in Sound: The Cognitive Psychology of Human Audition, Oxford Univ. Press. (1993) 146–198
23. Menon, V., Levitin, D.J., Smith, B.K., Lembke, A., Krasnow, B.D., Glazer, D., Glover, G.H., McAdams, S.: Neural Correlates of Timbre Change: In Harmonic Sounds NeuroI-mage, Vol. 17. (2002) 1742–1754
24. Beauchamp, J.: Synthesis by spectral amplitude and "Brightness" matching of analyzed musical instrument tones. J. of the Audio Engenering Society, Vol. 30(6). (1982) 396–406

25. McAdams, S., Winsberg, S., Donnadieu, S., De Soete, G., Krimphoff, J.: Perceptual scaling of synthesized musical timbres: Common dimensions, specificties, and latent subject classes. Psychol. Res. Vol. 58. (1995) 177–192
26. Grey, J. M.: Multidimensional perceptual scaling of musical timbres. J. Acoust. Soc. Am., Vol. 61. (1977) 1270–1277
27. Dubois, D.: Categories as acts of meaning: the case of categories in olfaction and audition. Cognitive Science Quarterly Vol. 1. (2000) 35–68
28. Castellengo, M.: Perception auditive des sons musicaux. In : Zenatti, A. (ed): Psychologie de la musique. PUF Paris (1994) 55–86
29. Dubois, D., Resche-Rigon, P., Tenin, A.: Des couleurs et des formes : catégories perceptives ou constructions cognitives. In : Dubois, D. (ed.): Catégorisation et cognition : de la perception au discours. Kim, Paris (1997) 7–40

The Clarinet Timbre as an Attribute of Expressiveness

Philippe Guillemain, Robin T. Helland, Richard Kronland-Martinet, and Sølvi Ystad

CNRS-Laboratoire de Mécanique et d'Acoustique
31, chemin Joseph Aiguier. 13402, Marseille cedex 20, France
guillem@lma.cnrs-mrs.fr

Abstract. In this paper we analyze clarinet sounds produced by a synthesis model that simulates the physical behavior of a real clarinet, in order to find a relationship between the clarinet timbre and the interpretation. Sounds have been obtained by varying two important control parameters of the synthesis model, namely the blowing pressure and the aperture of the reed channel. These parameters are also used to control real reed instruments. Four different timbre descriptors have further been applied to the sounds in order to investigate the timbre evolution as a function of these control parameters. The validity of the synthesis model has been verified thanks to an experimental setup with an artificial mouth, making it possible to generate and record sounds from a real clarinet while controlling the pressure and aperture of the reed channel. A relationship between the timbre and the physical behavior of the instrument has been found thanks to the physical synthesis model.

1 Introduction

The aim of this study is to use a physical synthesis model of the clarinet to get a better understanding of how a clarinet player controls the timbre during the play. For this purpose we use an existing synthesis model developed in our research group which closely simulates the physical behavior of a real instrument and which generates synthesized sounds that are extremely close to natural clarinet sounds. Initially the synthesis model was developed as a new tool for musicians and was adapted to a specific Yamaha WX5 controller that measures the aperture of the reed channel, the blowing pressure and the finger position. These parameters constitute the most important set of controls that a player uses on a real instrument. We have chosen to study the evolution of the timbre as a function of the aperture of the reed channel and the blowing pressure, in order to find out how the player controls the timbre of the instrument during the play. The advantage of using a synthesis model rather than a traditional instrument to study the timbre is that it gives access to calibrated and reproducible values of the control parameters and to a physical interpretation of the timbre behavior. Obviously the synthesis model has to be validated to make sure that it behaves like a traditional instrument with respect to the values of the physical control parameters. This was done by generating sounds from a real clarinet with an artificial mouth making it possible to calibrate the reed aperture and the blowing pressure like in the synthesis case. The sounds recorded from the real instrument globally depended on the control parameters in a similar manner as those from the synthesis model. This study also represents a starting point for future studies on interpretation, since the control device makes it possible to detect the values of the control parameters that a musician uses when playing this new instrument. This means that the instrument can be used as a tool to understand how the musician uses the timbre as a part of the interpretation.

The paper starts with a brief introduction of the physical model and its control parameters. In section 3, the timbre descriptors that were applied to the sounds are presented. Section 4 presents the results and links them with the physics.

2 Simplified Physical Model of a Clarinet

In this section, we briefly present the physical model used for the sound synthesis. It is made of three coupled parts. The first part is linear, and models the bore of the instrument using the impedance relation between the acoustic pressure and flow at the entrance of the resonator. The second part is nonlinear, based on the classical Bernoulli equation, and models the interaction between the acoustic velocity and the pressure differences between the mouth of the player and the entrance of the resonator. The acoustic flow is the product of the acoustic velocity with the aperture of the reed channel, which is linked with the reed displacement. The reed displacement is modeled as a pressure-driven single mode oscillator. In what follows, we use dimensionless variables for the pressure, flow, and reed displacement, according to [Kergomard, 1995].

2.1 Bore Model

The first linear part of the physical model corresponds to the resonator of the instrument. We here consider a highly simplified geometry made of a cylindrical bore. In particular, we neglect the role of the embouchure, the toneholes, the radiation losses and the register hole. Whatever the complexity of the bore geometry, its role on the sound generation process is fully determined by its input impedance. In the Fourier domain, the impedance links the acoustic pressure P_e and flow U_e in the mouthpiece. For the cylindrical geometry, assuming that the radiation impedance vanishes, the input impedance is classically written as:

$$Z_e(\omega) = \frac{P_e(\omega)}{U_e(\omega)} = i\tan(k(\omega)L) \qquad (1)$$

Here, U_e is normalized with respect to the characteristic impedance of the resonator: $Z_c = \frac{\rho c}{S_r}$, $S_r = \pi R^2$. R is the radius of the bore: $R = 7.10^{-3}$ in the clarinet case. This radius is large with respect to the boundary layers thicknesses and the wavenumber $k(\omega)$ is then expressed by: $k(\omega) = \frac{\omega}{c} - \frac{i^{3/2}}{2}\alpha c \omega^{1/2}$, where $\alpha = \frac{2}{Rc^{3/2}}\left(\sqrt{l_v} + \left(\frac{c_p}{c_v} - 1\right)\sqrt{l_t}\right)$. Typical values of the physical constants, in mKs units, are: $c = 340$, $l_v = 4.10^{-8}$, $l_t = 5.6.10^{-8}$, $\frac{C_p}{C_v} = 1.4$, $\rho = 1.3$.

2.2 Reed Model

We use a classical single mode reed model. It describes the displacement $x(t)$ of the reed with respect to its equilibrum point when it is submitted to an acoustic pressure $p_e(t)$:

$$\frac{1}{\omega_r^2}\frac{d^2x(t)}{dt^2} + \frac{q_r}{\omega_r}\frac{dx(t)}{dt} + x(t) = p_e(t) \tag{2}$$

where $\omega_r = 2\pi f_r$ and q_r are respectively the circular frequency and the quality factor of the reed. Typical values for these parameters are: $f_r = 2500Hz$ and $q_r = 0.2$.

In the Fourier domain, this last expression becomes:

$$X(\omega) = P_e(\omega)\frac{\omega_r^2}{\omega_r^2 - \omega^2 + i\omega q_r\omega_r}$$

which shows that the reed displacement is a low-pass and band-pass filtered version of the acoustic pressure.

2.3 Nonlinear Characteristics

The nonlinear characteristics is the most important part of the model, since it is the engine of the self-oscillations production. The simple model used here is based on the stationary Bernouilli equation that links in a nonlinear way the acoustic velocity with the pressure difference between the bore and the mouth of the player. The acoustic flow entering the bore is then the product between the reed channel opening and the acoustic velocity. By using dimensionless variables, the acoustic pressure $p_e(t)$, the acoustic flow $u_e(t)$ and the reed displacement $x(t)$ are linked in a nonlinear way at the input of the resonator as follows:

$$u_e(t) = \frac{\zeta}{2}(1 + sign(1 - \gamma + x(t)))sign(\gamma - p_e(t))(1 - \gamma + x(t))\sqrt{|\gamma - p_e(t)|} \tag{3}$$

The parameter ζ characterizes the whole embouchure and takes into account the lip position and the section ratio between the mouthpiece opening and the resonator: $\zeta = \sqrt{H}\sqrt{\frac{2\rho}{\mu_r}}\frac{cw}{S_r\omega_r}$, where μ_r and w respectively denote the mass per unit length and the width of the reed. ζ is proportional to the square root of the reed position at equilibrium H and represents an important control parameter on which the player can act. Common values of ζ for the clarinet are between 0.2 and 0.6.

The parameter γ is the ratio between the pressure inside the mouth of the player and the pressure for which the reed closes the embouchure in the static case: $\gamma = \frac{p_m}{p_M}$, where $p_M = H\omega_r^2\mu_r$ is the static beating reed pressure. In a lossless bore model, γ evolves from $\frac{1}{3}$ which is the oscillation step, to $\frac{1}{2}$ which corresponds to the position at which the reed starts beating.

The parameters ζ and γ are the most important continuous performance parameters since they respectively represent the way the player holds the reed and the blowing pressure inside the instrument.

Figure 1 represents the non linear characteristics of the reed for the limit case $\omega_r = \infty$ ($\zeta = 0.3, \gamma = 0.45$). In this case, the displacement $x(t)$ of the reed reduces to the acoustic pressure $p_e(t)$ itself: the reed activity can be considered as a single spring. As we shall see in the next sections, such a situation occurs when the frequency support of the acoustic pressure remains smaller than the reed resonance frequency. This plot shows one discontinuity at its left side, corresponding to a cancellation of the acoustic flow for $p_e = x = \gamma - 1$ when the reed closes completely the input of the bore, and one infinite derivative at its right side for $p_e = \gamma$ corresponding to the limit between a positive and negative acoustic flow.

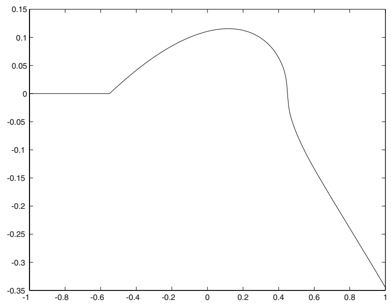


Fig 1: *Nonlinear characteristics of the reed (u_e as function of p_e)*

2.4 Coupling of the Reed and the Resonator

Combining the impedance relation, the reed displacement and the nonlinear characteristics, the acoustic pressure, acoustic flow and reed displacement in the mouthpiece can finally be found by solving the following set of equations:

$$\frac{1}{\omega_r^2}\frac{d^2x(t)}{dt^2} + \frac{q_r}{\omega_r}\frac{dx(t)}{dt} + x(t) = p_e(t) \tag{4}$$

$$P_e(\omega) = Z_e(\omega)U_e(\omega) = i\tan\left(\frac{\omega L}{c} - \frac{i^{3/2}}{2}\alpha c\omega^{1/2}L\right)U_e(\omega) \tag{5}$$

$$u_e(t) = \mathcal{F}(p_e(t), x(t)) \tag{6}$$

The flow diagram corresponding to this system of three coupled equations, in which the reed and the non-linearity are introduced as a nonlinear loop linking the output p_e to the input u_e of the resonator, is shown in figure 2.

From the pressure and the flow inside the resonator at the moutpiece level, the external pressure is calculated by the relation: $p_{ext}(t) = \frac{d}{dt}(p_e(t) + u_e(t))$. This expression corresponds to the simplest approximation of a monopolar radiation.

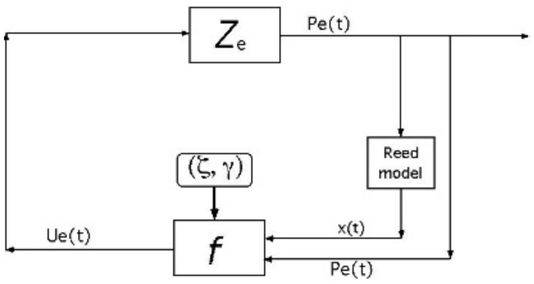


Fig 2: *Nonlinear Synthesis Model*

The digital transcription of equations (4,5,6) and the computation scheme that explicitely solve this coupled system is achieved according to the method described in [Guillemain, 2002].

3 Caracterisation of Timbre by Classical Descriptors

Timbre is a component in auditory perception which has been studied and described by several authors [McAdams, 1995][Menon, 2002][Grey, 1977][Jensen, 1999]. It is probably the most important feature of auditory events, and yet we do not have a precise definition or a complete understanding of timbre. It is often described by an "anti-definition" commonly referred to as those aspects of sound quality other than pitch, loudness, perceived duration, spatial location, and reverberant environment [McAdams, 1993]. For this reason, we don't take into consideration the loudness in this study, though this attribute is obviously an important part of the interpretation. More concretely, timbre can be considered as the principal feature that allows us to distinguish different instruments playing the same note. It is generally assumed to be multidimensional and is often described by a 3 dimensional space. [Grey, 1977] defines the 3 dimensions as spectral centroid, attack time and the irregularity of the spectrum, while [McAdams, 1995] in some studies uses the mean value of the variation of the spectral components as a function of time (spectral flux) as a descriptor for the third dimension. Authors seem to agree that the spectral centroid and the attack time are important timbre descriptors, while the last dimension is more ambiguous. As mentioned in the introduction, the aim of this study is to find a relationship between the timbre and the physical behavior of the instrument in order to get a better understanding of the relationship between timbre and interpretation. The sounds that are used in this experiment are generated by the synthesis model based on a physical model simulating sounds close to natural clarinet sounds and suimmarized in section 2. This model has been constructed to propose a new tool for musicians and can be piloted by a wind-like instrument controller. Since we restrict ourselves to the use of two parameters to control the sound (reed aperture, blowing pressure), we have chosen to study the timbre evolution for different values of these control parameters for a sustained sound with a fixed length of resonator. Actually, these two control parameters are explicitely related to the parameters ζ and γ described in section 2. The values of the control parameters were chosen within a "reasonable" range, meaning that we only considered sounds that corresponded to a common play.

In the clarinet case where the amplitudes of the spectral components vary a lot, especially the ratio between odd and even harmonics, the spectral irregularity which represents a measure of the amplitude differences between successive components seems to be important [Jensen, 1999]. Since we restrain this study to sustained sounds, the spectral flux will not be important since the spectral centroid remains constant during the stable part of the sound. Thus, as a first approach we have studied the evolution of the attack time, the spectral centroid and the spectral irregularity of the synthesized clarinet sounds with respect to the physical parameters. The spectral centroid, which is related to the brightness of a sound, can be calculated from the formulae [Beauchamps, 1982]:

$$SC = \frac{\sum kA_k}{\sum A_k}$$

where k is the index of a spectral component and A_k is the amplitude value of the k^{th} component.

The spectral centroid has been calculated by convolution of the time varying signal with a hanning window of size 1024 samples, with 50% overlap between frames. For each frame the centroid is calculated up to the frequency $22050Hz$. For each clarinet sound, a centroid value has been obtained by calculating the mean value of the time-evolving centroid over a duration corresponding to the second half of the sound (from 0.5 s to 1s), so that the steady state is reached for all sounds.

The global attack time AT was found by calculating the time elapsed for the amplitude of the clarinet sound to increase from 10% to 90% of its maximum absolute value. The individual attack times of the five first harmonics were also calculated, but these were not found to be significant compared to the global attack time.

The spectral irregularity (IRR), which is a measure for the amplitude difference between successive components and which is highly important for clarinet sounds since it represents the differences between odd and even components, was obtained from the expression [Jensen, 1999]:

$$IRR = \frac{\sum (A_k - A_{k+1})^2}{\sum A_k^2}$$

As already mentioned, traditional timbre descriptors are often insufficient to describe the subtle timbre variations of the same instrument, and are often more similar for sounds from different instruments with the same pitch than for sounds from the same instrument with a different pitch [Jensen, 1999]. As we shall see in section 4 where the results from the analysis are described, this observation is confirmed by our approach. It was therefore necessary to look for additional criteria that would make it possible to distinguish sounds with similar attack times, spectral centroids and spectral irregularities, but different timbre qualities. For this purpose we have proposed a descriptor related to the spectral bandwidth (SBW). This descriptor is linked to the spreading of the spectral components and corresponds to the standard deviation of the curve defined by the energy envelope of the symmetric spectrum for negative and positive frequency values. This avoids the SBW to be correlated with the spectral centroid.

4 Results

In this section we describe the results of the analysis of the clarinet sounds generated by the physical synthesis model described in section 2. As already mentioned we have chosen to vary two control parameters (ζ and γ) linked to the blowing pressure and the reed aperture. Hence, 100 clarinet sounds generated with the same bore length (L=0.5) (fundamental frequency $\simeq$ 171Hz) have been obtained by varying γ from 0.4 to 0.5 (10 different values) and ζ from 0.2 to 0.5 (10 different values). We first give a description of the timbre variations related to the different sounds. Then we give a physical interpretation of the observed results.

4.1 Timbre Variations as a Function of the Control Parameters

In this section we show how different timbre descriptors vary with the reed aperture and the blowing pressure. We have calculated the four timbre descriptors for the hundred clarinet sounds; namely the attack time, the spectral centroid, the spectral irregularity and the spectral bandwidth. Figures 3 to 6 show the descriptors as a function of ζ and γ. The two control parameters seem to have similar effects on the timbre descriptors. Figure 3 shows that increasing ζ and γ increases the values of the spectral centroid. We also see that in several cases the centroid is the same for different sounds (i.e. different combinations of parameters). For low values of ζ and γ, these sounds are perceptually similar, while they are different for higher values. This might indicate that the spectral centroid is a sufficient descriptor for small parameter values while other descriptors have to be added for higher parameter values. The spectral centroid does not change significantly for low values of ζ and γ, but when the two parameters reach a given value, the centroid exhibits a steep increase. This is seen best in the contour plot, Fig 3, where the close lines indicate a sudden rise in the centroid value. Also, the pressure parameter is found to have a stronger effect on the centroid, giving a larger global increase than the aperture parameter. For low values of the control parameters the attack time changes very abrubtly, whereas the attack time is stable for higher values. The contour plot shows that almost half of the synthesis sounds have an attack time within a 50ms region, which makes the attack time a rather insufficiant descriptor when high parameter values are used. The spectral irregularity is the only descriptor that does not act monotonously with increasing ζ and γ. As seen from the 3D graph, an increase in blowing pressure causes an increase in spectral irregularity, provided that the reed aperture is adequately low. Whereas for ζ values above 0.3, an increase of pressure causes a slight decrease in irregularity. The overall change in irregularity is more influenced by ζ than γ, which can be seen from the contour plot, where the contour lines are partly parallell with the pressure axes.

The results of these descriptors give us an indication of how the different control parameters influence the timbre. However, for high values of ζ and γ there is a need to apply an additional descriptor, since the attack time is almost constant and the spectral irregularity does not change significantly. Sounds with different control parameter values in the higher regions, which have the same mean spectral centroid, are found to be perceptually different. We have therefore proposed a fourth timbre descriptor that we have called the spectral bandwidth describing the spreading of the spectral energy. From 3D representations we see that the spectral bandwidth is useful as an additional descriptor to seperate sounds of equal mean spectral centroid, and give a perceptual description of the difference between these sounds. The spectral bandwidth rises steadily with increasing values of ζ and γ, meaning that the energy in high frequency components increases. Fig 3 also show that the reed aperture has a considerably larger influence on the bandwidth than the pressure, as opposed to the spectral centroid where the blowing pressure has the most influence.

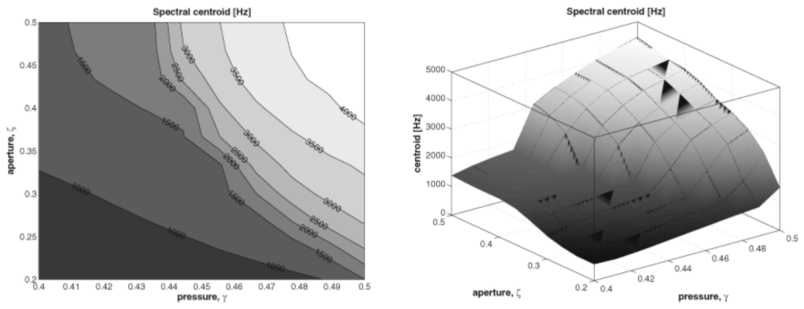


Fig 3: *Contour plot and 3D plot of the mean spectral centroid as it evolves for different values of blowing pressure and reed aperture. The values of γ and ζ evolve respectively from 0.4 to 0.5 and from 0.2 to 0.5.*

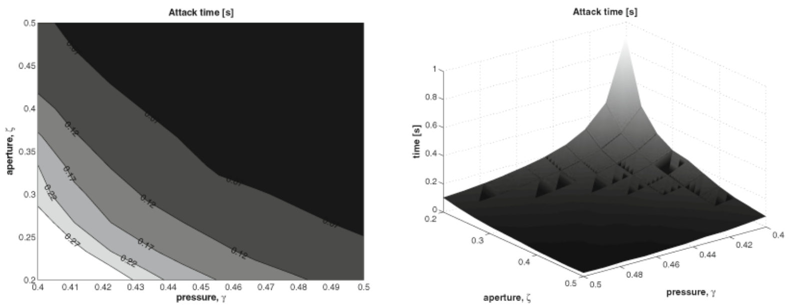


Fig 4: *Contour plot and 3D plot of the global attack times for the hundred clarinet sounds, for different values of pressure and aperture.*

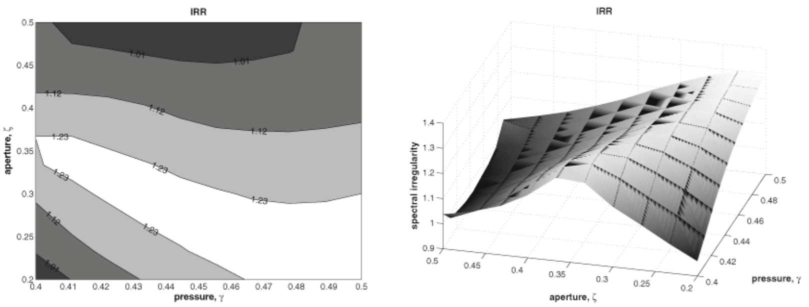


Fig 5: *Contour plot and 3D plot of the spectral irregularity, for different values of pressure and aperture.*

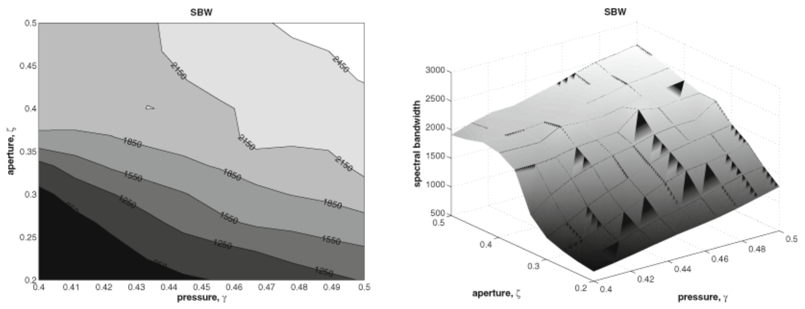


Fig 6: *Contour plot and 3D plot of the spectral bandwidth, for different values of pressure and aperture*

4.2 Physical Interpretation of the Timbre Variations

In this section, we consider four different situations, each corresponding to a particular combination of the control parameters ζ and γ. The first situation (γ small, ζ small) corresponds to figures (7,11,15). The second situation (γ small, ζ large) corresponds to figures (8,12,16). The third situation (γ large, ζ small) corresponds to figures (9,13,17). The forth situation (γ large, ζ large) corresponds to figures (10,14,18).

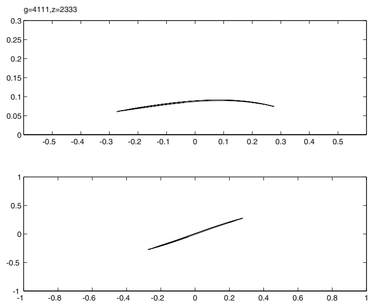
Several physical phenomena explain the evolution of the amplitude of the harmonics of the clarinet spectrum. For small values of the two control parameters ζ and γ, the non-linear mechanisms are responsible for the production of harmonics. Figure 7 (top, resp. bottom) shows the internal flow (resp. reed displacement) as a function of the internal pressure. This plot can be directly linked to figure 1 describing the non-linear characteristics of the reed. For low values of ζ and γ, the reed behaves like a simple spring and moves proportionally to the pressure (see figures 11,15). In this case the internal pressure and flow have a frequency support smaller than the resonance frequency of the reed.

For small values of the blowing pressure (parameter γ) and increasing values of the reed aperture (parameter ζ) (corresponding to figure 8, 12, 16), the resonance frequency of the reed causes a "formant" to appear in the sound. In this case, the frequency spreading of the acoustic pressure is higher than the reed resonance frequency and the reed acts as a pressure amplifier around its resonance frequency. This can be seen on figure 8 by the hysteresis effect showing that the flow and reed displacement do not follow the same path when the reed opens and closes, and on the flow spectrum on figure 16.

For high values of γ and small values of ζ (figure 9,13,17) the important displacement of the reed provoked by the pressure causes the reed to beat against the table. This brutally cancels the flow (figure 13) and introduces a singularity in the sound creating a sudden increase in its brightness. For a given γ value the collision between the reed and the table happens for a particular ζ value, causing a sudden increase in the brightness of the sound.

For high values of both γ and ζ (figures 10, 14 18), the different mechanisms observed in the previous cases appear simultaneously. Moreover, the important variation of the flow compared to the small variation of the pressure when the reed is completely open causes the spectrum to gain even more components (the right part of the top curve of figure 10). Indeed, for $p_e(t) = \gamma$, the derivative of the non-linear characteristics is infinite and a very small variation of pressure causes an abrupt variation of flow.

Whatever the values of the pressure and reed aperture parameters, since the input impedance of the bore mainly contains odd harmonic peaks, the internal pressure $p_e(t)$ contains very few even harmonics and always looks like a square signal. This is visible on figures 15,16,17,18. The even harmonics come from the internal acoustic flow $u_e(t)$ and are produced by the nonlinearity, from the term: $\sqrt{\gamma - p_e(t)}$. This nonlinearity explains why the level of even harmonics in the flow increases when the difference in pressures between the blowing pressure and the bore pressure (and hence the blowing pressure) also increases. Since the external pressure is expressed as the time derivative of the sum of the internal acoustic flow and pressure, the rate of its even harmonics increases with respect to γ, and hence the spectral irregularity decreases.



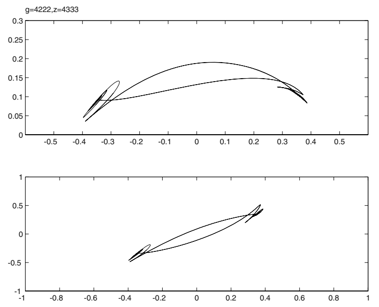
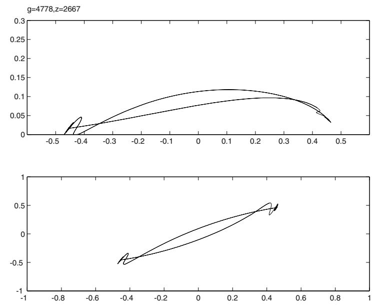


Fig 7(left),8(right): *internal acoustic flow (upper part of the figure) and reed displacement (lower part of the figure) as a function of the pressure for two different values of the control parameters γ and ζ. Figure 7 corresponds to γ=0.4111 and ζ=0.2333 while figure 8 corresponds to γ=0.4222 and ζ=0.4333.*



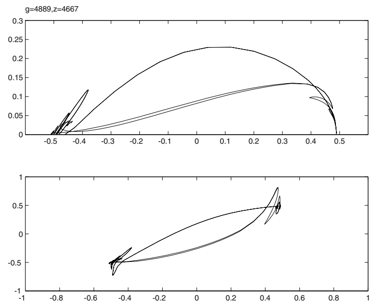
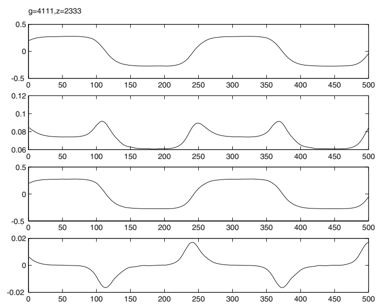


Fig 9(left),10(right): *internal acoustic flow (upper part of the figure) and reed displacement (lower part of the figure) as a function of the pressure for two different values of the control parameters γ and ζ. Figure 9 corresponds to γ=0.4778 and ζ=0.2667 while figure 10 corresponds to γ=0.4889 and ζ=0.4667.*



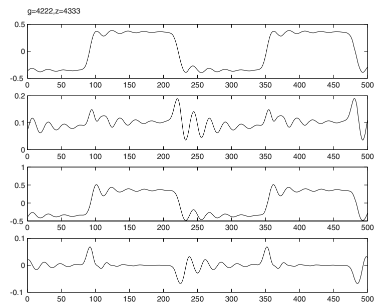
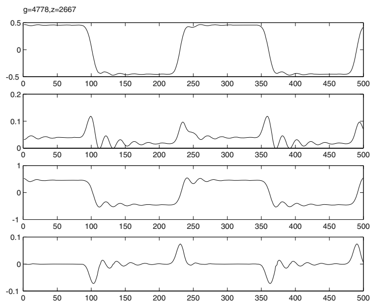


Fig 11(left),12(right): *from top to bottom :internal acoustic flow $p_e(t)$, acoustic flow $u_e(t)$, reed displacement $x(t)$ and external pressure $p_{ext}(t)$. Figure 11 corresponds to γ=0.4111 and ζ=0.2333 while figure 12 corresponds to γ=0.4222 and ζ=0.4333. Horizontal axis in samples.*



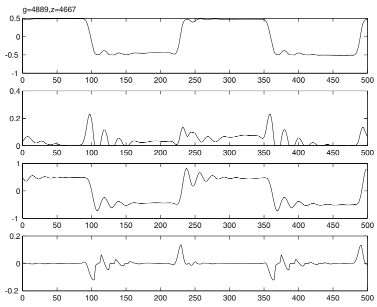
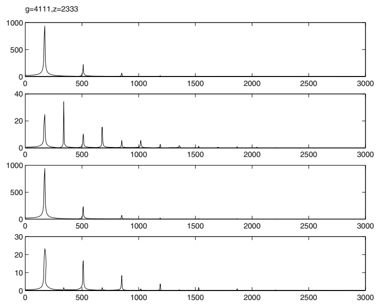


Fig 13(left),14(right): *from top to bottom :internal acoustic flow $p_e(t)$, acoustic flow $u_e(t)$, reed displacement $x(t)$ and external pressure $p_{ext}(t)$. Figure 13 corresponds to γ=0.4778 and ζ=0.2667 while figure 14 corresponds to γ=0.4889 and ζ=0.4667. Horizontal axis in samples.*



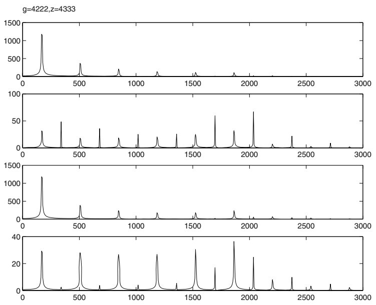
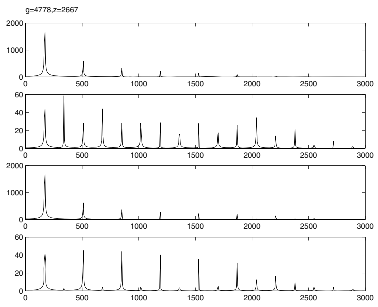


Fig 15(left),16(right): *from top to bottom :internal acoustic flow* $p_e((\omega)$*, acoustic flow* $u_e((\omega)$*, reed displacement* $x((\omega)$ *and external pressure* $p_{ext}(\omega)$*. Figure 15 corresponds to* γ*=0.4111 and* ζ*=0.2333 while figure 16 corresponds to* γ*=0.4222 and* ζ*=0.4333. Horizontal axis in Hertz.*



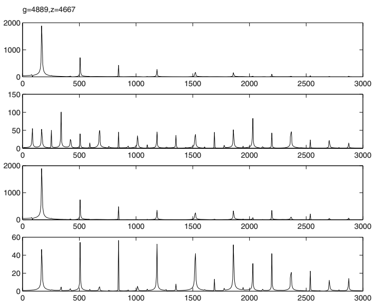


Fig 17(left),18(right): *from top to bottom :internal acoustic flow* $p_e((\omega)$*, acoustic flow* $u_e((\omega)$*, reed displacement* $x((\omega)$ *and external pressure* $p_{ext}(\omega)$*. Figure 17 corresponds to* γ*=0.4778 and* ζ*=0.2667 while figure 18 corresponds to* γ*=0.4889 and* ζ*=0.4667. Horizontal axis in Hertz.*

5 Conclusion

Musical interpretation is an extremely complicated process aiming at generating emotions to the listener. An interesting question we started to address is how the timbre variations of instruments are controlled in such a context. For that, we focused on the clarinet timbre for which accurate physical models running in real time can be designed. This allowed us to systematically study the way the timbre of this instrument varies with respect to two important control parameters: the blowing pressure and the reed aperture, both of them being controllable by the player. Four timbre descriptors have been studied for this purpose: the spectral centroid, the attack time, the spectral irregularity and the spectral bandwidth. All of these descriptors act quasi-monotonously with each control parameter. This is probably one of the reasons why this instrument allows an intuitive control of the sound. Another consequence is that given characteristics of the timbre can be obtained in different ways by compensating the influence from one of the control parameters by the other one. This allows for example the brightness of the sound to be kept constant while changing its richness. Such possibilities probably are clues to a better understanding of the notion of playability and of the musical interest of an instrument. Actually, they are widely used by musicians during a musical performance. Thanks to the use of a realistic physical synthesis model, one then discussed the relationship between the physics of the instrument and the timbre. This is an important step towards a better understanding of what part of the instrument design is allowing subtle sound variations. Even though this is a preliminary study, we have shown the importance of the two regimes defined by the free motion and the beating of the reed on the table of the mouthpiece. These regimes are obtained for control parameters depending on the mean aperture and the stiffness of the reed, showing the importance of the design of the embouchure together with a good choice of the reed. Another important aspect of the timbre is linked to the first eigenfrequency of the reed. Actually, one has shown how the reed acts on the sound spectrum, increasing the brightness by accentuating the spectral peaks around the first reed eigenfrequency. The quality factor of the reed also seems to be of importance for the resulting sound since it acts on the spectral bandwidth altered by the reed. These conclusions show the close relationship between the clarinet, the playing and the timbre of the produced sounds. This leads to interesting possibilities of musical interpretation as soon as the physical design of the instrument is adequate in the sense that the control is easy enough and intuitive. Rather than answering the numerous questions linked to instruments and playing, this study has opened new fields of investigation. Indeed, an experimental setup using an artificial mouth connected to a real clarinet made it possible to perfectly control pressure and aperture of the real instrument just like we did with the synthesis model. This experiment made it possible to validate the synthesis model and confirmed its close behavior to a real instrument. This comforted us in our choice of using the physical synthesis model in our work towards a better understanding of the rules linking interpretation and timbre variations.

References

[McAdams, 1995] McAdams, S., Winsberg, S., Donnadieu, S., De Soete, G., and Krimphoff, J. 1995. Perceptual scaling of synthesized musical timbres: Common dimensions, specificities, and latent subject classes. Psychol. Res. 58: 177-192.

[McAdams, 1993] McAdams, S. 1993. Recognition of sound sources and events. In Thinking in Sound: The Cognitive Psychology of Human Audition (S. McAdams and E. Bigand, Eds.), pp. 146-198. Oxford Univ. Press, Oxford.

[Beauchamp, 1982] Beauchamp J., 1982, Synthesis by spectral amplitude and "Brightness" matching of analyzed musical instrument tones. J. Acoust. Eng. Soc., 30(6): 396-406.

[Grey, 1977] Grey, J. M. 1977. Multidimensional perceptual scaling of musical timbres. J. Acoust. Soc. Am. 61: 1270-1277.

[Guillemain, 2002] Ph. Guillemain, J. Kergomard, Th. Voinier, “Procédé de simulation et de synthèse numérique d’un phénomène oscillant”, french patent request n0213682, Oct 2002.

[Guillemain, 2003] Ph. Guillemain, Th. Voinier, “Characterization of musical performance using Physical Sound Synthesis Models”, Lecture Notes on Computer Sciences, Vol. 2771, pp 64-73, Kock-Wiil Ed.,Springer-Verlag, (2003)

[Jensen, 1999] Jensen, K., 1999, Timbre models of musical sounds, Ph. D., DIKU press, Copenhagen, Denmark.

[Kergomard, 1995] Kergomard J. “Elementary considerations on reed-instruments oscillations”, in Mechanics of Musical Instruments, Lectures notes CISM, Springer. (1995).

[Menon] V. Menon,, D. J. Levitin, B. K. Smith, A. Lembke,B. D. Krasnow, D. Glazer, G. H. Glover, and S. McAdams, Neural Correlates of Timbre Change in Harmonic Sounds NeuroImage 17, 1742-1754 (2002)

A Graph Theoretic Approach to Melodic Similarity

Goffredo Haus[1] and Alberto Pinto[2]

[1] LIM - Laboratorio di Informatica Musicale,
DICO - Dipartimento di Informatica e Comunicazione
Università degli Studi di Milano
haus@dico.unimi.it

[2] LIM - Laboratorio di Informatica Musicale,
DICO - Dipartimento di Informatica e Comunicazione
Università degli Studi di Milano
pinto@pcteor1.mi.infn.it

Abstract. Common music information retrieval methods are based upon editing distances, reductionism or functional analysis tecniques. We adopt an approach which looks into a thematic fragment *(TF)* globally. This leads to associate a *musical graph* to each TF which preserves its more abstract content. Then, necessary conditions for graph inclusion are introduced and we give a similarity function between graphs which allows to assign different weights to the elements belonging to different graph powers. The advantage is that graphs catch more musical transformations than other methods, like permutations of subfragments.

Keywords: melodic similarity, musical graph, graph metric, similarity function, eulerianity, hamiltonicity, inclusion.

1 Introduction

The most usual queries on musical files are title, author, year of publishing, etc. However, one may be interested in melodies containing a given sequence of notes or sequences 'similar' to a given one: this is called *music information retrieval (MIR).*

The main result of our work is a new and useful model that allows the retrieval of music contents. In fact it is evident that the increase of files size and number makes their teatment like pure sequences of bytes quite impossible. On the contrary, it is necessary to consider the peculiar characteristic of the musical contest, in other words we are in need of a *model.* This allows a more precise and fast retrieval and facilitates the recognition of some particular transformation the theme could have undergone, like traspositions or inversions.

We are first going to analyse two approaches known in literature: Lerdahl & Jackendoff's grammars and Tenney & Polansky's metrics. Then we will propose a new approach based on graph theory.

Given a tematic fragment (TF) M on n distinct notes and of lenght m, we will describe the construction of a *musical graph* representing M, a weighted eulerian oriented multigraph $G(M) = (V_G, A_G, \partial_0, \partial_1, d, p)$ with n vertices and m arrows, where V_G has a metric space structure (V_G, d).

Afterwards we define the concept of *similarity* and give some *necessary conditions* for the inclusion of TFs by *graph invariants* and *metrics* on V_G.

2 Related Work

Here we are going to overview the approaches we have prior analised and that we consider more relevant in the literature on *music information retrieval.* Substantially they can be divided into two classes:

1. reductionistic models
2. computational models

The first category leads to reduce musical information into some "primitive types" and then to compare the reduced fragments. The second category is based upon algorithms which utilize the whole sequences of notes. There isn't a clear separation between the two classes. In addition we can say that some methods cannot be clearly ascribable to the first or second class.

The Psychoperceptive Approach A very interesting reductionistic approach refers to Fred Lerdahl e Ray Jackendoff's studies. In 1983 Lerdahl e Jakendoff ([4]) published their researches oriented towards a *formal description of the musical intuitions of a listener who is experienced in a musical idiom.* Their work wasn't directly related to *MIR*, their purpose was the development of a formal grammar which could be used to analise any tonal composition. However, in case of tematic fragments (TFs), it would be possible to reduce the TFs into "primitive types", showing formal similarities according to the defined grammar.

The aim is to describe, in a simplified manner, the *analitic system of the listener*, i.e. rules which allow the listener to segmentates and organizes a gerarchy of musical events. On this basis, *score reductions* are applied, gradually deleting the less significant events. In this way we can obtain a simplification of the score and in the meantime we can preserve sufficient information which allow to mantain recognizability.

The study of these mechanisms, which can be either conscious and inconsious, allows the construction of a *formal grammar* able to describe the fundamental rules followed by human mind in the recognition of the underlying structures of a musical piece.

The grammar construction is very complex and a complete treatment of the problem goes beyond our purposes.

Functional Metrics The approach proposed by Tenney and Polansky ([1], [2]) founded on the concept of *morphology*, substantially a finite sequence of comparable elements.

In this contest melodies are finite sequences, i.e. functions defined on a finite subset of $\mathbf{N}$ and whose values are in mono ($\mathbf{Q}$) or n-dimensional ($\mathbf{Q}^n$) spaces, according to the characterizing parameters.

Thus, distance concepts are those erhedited by metrics defined on (discrete) functional spaces.

The measure of those distances involves the creation, the recognition and the analysis of variations and trasformation of morphological parameters like pitches, onsets, harmonic relations, sequences of timbre related values and, more generally, any kind of observable related to melody.

As pointed out by Tenney [1], we have to distinguish between statistical and morphological properties. Generally, statistical are *global* and *time independent*, like the mean value or standard deviation of a parameter, while morphological characteristics are described by the *'profile'* of parameters and depend on elements ordering. It is also possible to use statistic measures of parametric profile as parameters at a higher level (gerarchy of profiles); in this way we can analyse the melodic profile at different levels by the application of different metrics.

Definition 1. *A morphology is an ordered set of elements belonging to the same ordered set M. The elements in M are identified by M_i , $i = 1, \cdots, L$; $L = |M|$.*

Some examples of morphologies are pitch sequences, rhythm sequences, harmonic sequences, etc. (cfr. [1], [2]).

'Distance' and 'similarity' used in those contests aren't well-defined in respect of the mathematical notion of metric.

Definition 2. *Given a set S, a function*

$$d : S \times S \longrightarrow \mathbf{R} \tag{1}$$

is a distance function *or a* metric *if* $\forall a, b, c \in S$ *holds:*

1. $d(a, b) \geq 0$
2. $d(a, b) = 0$ *iff* $a = b$
3. $d(a, b) = d(b, a)$
4. $d(a, b) \leq d(a, c) + d(c, b)$.

(S, d) *is called a* metric space.

Morphological metrics are metrics on *morphologies* (metrics on ordered sets).

Metrics on spaces of real valued functions are useful models for morphological metrics. For example, given two continuous real valued functions $f(t)$ e $g(t)$, defined on $[m, n]$, there are two intuitive amplitude metrics:

$$d(f, g) = \sup\{|f(t) - g(t)|\} \text{ (sup-metric)} \tag{2}$$

and

$$d(f,g) = \int_n^m |f(t) - g(t)|dt \ \text{ (amplitude metric)} \tag{3}$$

Working in discrete spaces, the integrals will be replaced by sums.

By replacing $f(t)$ and $g(t)$ with their derivatives of any order, Tenney and Polansky obtained metrics by measuring the mean amplitude difference of the corresponding rate of changement of the two functions. For discrete functions (like morphologies) the derivative is substituted by the *difference function* of first (second, third, $\cdots$) order.

Polansky ([2],[3]) analises in detail a lot of interesting functional metrics, ordered and non ordered, with some applications to melodic sequences.

However this approach is, in our opinion, quite distant from the music itself. We think that it is not so natural to consider melodies in the same way of functions; in fact a large part of musical information are irreduceble to the concept of function.

In the next section we are going to introduce a new kind of approach which we consider much more connected to the musical facts, though it uses tipically mathematic concepts of graph theory.

3 Background

In this section we'll recall some graph-theoretic definitions and results which will be used later. We'll suppose the reader knows the elementary notions of graph theory (cfr. for example [7], [8]), particularly the notions of vertex degree, graph inclusion, eulerianity, hamiltonicity and spanning tree.

In this paper we'll consider quite only eulerian connected oriented multigraph defined on finite sets. To fix the notation, now we give some definition.

Definition 3. *A* finite oriented (multi) graph $G = \{V_G, A_G, \partial_0, \partial_1\}$ *is a set* V_G *of objects* v_i*,* $i \in I = \{1, 2, ..., n\}$ *(the* vertices*), a set* A_G *of arrows* a_i*,* $i \in I = \{1, 2, ..., m\}$ *and a couple of functions* ∂_0*,* ∂_1 *such that* $\partial_0 : A_G \rightarrow V_G$ *e* $\partial_1 : A_G \rightarrow V_G$*.*

Definition 4. *A* graph morphism $f : G \rightarrow G'$ *is a couple of functions* $f_A : A_G \rightarrow A_{G'}$ *and* $f_V : V_G \rightarrow V_{G'}$ *which make the diagram below to commute.*

$$\begin{tikzcd} A_G \arrow[r, "f_A"] \arrow[d, "\partial_0, \partial_1"'] & A_{G'} \arrow[d, "\partial'_0, \partial'_1"] \\ V_G \arrow[r, "f_V"'] & V_{G'} \end{tikzcd} \tag{4}$$

Definition 5. *Given an oriented graph G, the* opposite graph G^{op} *is the graph obtained from G reversing the orientation of all the arrows.*

3.1 Inclusion

Now we will introduce a notion of inclusion, different from the standard one.

Definition 6. *A graph G is* included *in a graph G' if $V_G \subseteq V_{G'}$ and exists a partition $\mathcal{P}(A_{G'})$ of the arrows set of G', $A_{G'}$, in trails such that the diagram*

$$\begin{tikzcd} A_G \arrow[r, tail, "i_A"] \arrow[d, tail, "\partial_0, \partial_1"'] & \mathcal{P}(A_{G'}) \arrow[d, tail, "\partial'_0, \partial'_1"] \\ V_G \arrow[r, tail, "i_V"'] & V_{G'} \end{tikzcd} \tag{5}$$

commute; where $i = (i_A, i_V)$ is the usual graph inclusion.

Remark 1. If the partition is *the finest* one, i.e. all classes are singleton, the definition collapse to the standard inclusion.

Remark 2. The partition may not be unique.

Proposition 1. *The inclusion defined above is a partial order relation.*

Proof. Reflexivity and transitivity are obvious. Using the injectivity of i and of its inverse, $V_G = V_{G'}$ and $A_G = A_{G'}$; so, by the commutativity of diagram (5), we've proved the antisimmetry.

Remark 3. The inclusion defined above works with labelled graphs. It may be useful enlarging this concept to the situation where V_G is 'quite' included in $V_{G'}$. In fact in musical graphs is not relevant to preserve the labels because we study objects invariant by musical tranformations (i.e. permutation of labels preserving the metric structure defined on V_G).

So we will give the next definition.

Definition 7. *A graph G is* weakly included *in a graph H iff exists a subgraph G' of H isomorphic to G.*

$$G \cong G' \subseteq H \tag{6}$$

3.2 Union

Let's consider a couple of graphs G and H.

Definition 8. *We define the* union *of G and H the graph $G \sqcup H$ such that $V_{G \sqcup H} = V_G \sqcup V_H$ and $A_{G \sqcup H} = A_G \sqcup A_H$.*

Remark 4. If G and H are connected eulerian graphs, it may be convenient assuming $V_G \sqcap V_H \neq \emptyset$. Doing so the result is equally connected and eulerian.

Proposition 2. *The union operator defines the minimum superior bound $G \vee H$ for every couple of graphs G and H in respect to the relation of (strong) inclusion.*

Proof. If a graph I containing G and H is contained in $G \vee H$, it has to be $V_I = V_{G \vee H}$ because also hold $V_G \subseteq V_I$, $V_H \subseteq V_I$ and $V_I \subseteq V_{G \vee H}$.

By the arc axiom, should be $|A_G| + |A_H| \leq |A_I| \leq |A_{G \vee H}| = |A_G| + |A_H|$. Then $I = G \vee H$.

3.3 Graph Complexity

Definition 9. *The* complexity *$k(G)$ of a graph G is the number of equally oriented spanning trees of G.*

The following three propositions evidence the importance of graph complexity in our model. Their importance will be more clear in the next section.

Proposition 3. *Let H be the graph obtained from a graph G substituing an arrow $a : v_i \rightarrow v_j$ with a couple of arrows $a_1 : v_i \rightarrow v_k$ and $a_2 : v_k \rightarrow v_j$, with $v_i, v_j, v_k \in V_G$. Then*

$$k(H) \geq k(G). \tag{7}$$

Proof. Let T be a spanning tree which contains the arrow $a_1 : v_i \rightarrow v_k$. If we replace a with the trail $a_1 a_2$, the result from T with the substitution of a by a_1 or by a_2 is yet a spanning tree; so the complexity of G can only increase.

Proposition 4. *Let H be the graph obtained from a graph G of order $n(G)$ adding a vertex v_{n+1} to V_G and replacing an arrow $a : v_i \rightarrow v_j$ with the couple of arrows $a_1 : v_i \rightarrow v_{n+1}$ and $a_2 : v_{n+1} \rightarrow v_j$. Then*

$$k(H) \geq k(G). \tag{8}$$

Proof. We can divide the spanning trees of G in two classes: X={spanning trees containing a} and Y={spanning trees which do not contain a}. Obviously we have $k(G) = |X| + |Y|$. If $\alpha \in Y$ then $\alpha \vee a_2$ is a spanning tree of H. Thus the complexity of H is at least $|Y|$. Let's consider a tree $\beta \in X$. Replacing a by the trail $a_1 a_2$ we obtain yet spanning tree. Finally we have $k(H) \geq |Y| + |X| = k(G)$.

Proposition 5. *Graph complexity is a monotonic function respect of the order relation of graph inclusion previously defined.*

Proof. Let G be strongly contained in H. Then, if $V_G \subset V_H$, the corollary 3 implies that $k(G) \leq k(H)$ and if $V_G \leq V_H$, the corollary 4 implies $k(G) = k(H)$. Now let G be included weakly in H. The invariance of complexity under isomorphism and the previous case proves the theorem.

4 The Model

Our principal purpose was to look into a theme fragment globally, not only as a pure sequence of notes. So that's why we've structured musical information in a different way, but preserving the more abstract content.

4.1 Musical Graphs

Now we are going to describe how to build a graph model of a TF. We consider a structured set of TF (a *database*) and a TF (the *query*) which has to be compared with every set-element. This is how to proceed:

1. build a representative graph for every TF
2. work with graphs instead of TF.

In this way, we can recognise a greater number of relevant musical similarities and we can reduce the number of TF wich should be compared. Moreover, TFs of the archive which have the same representative graph can be identified yet.

Let M be a TF of length $m = |M|$ and consider three characterizing sequences of observable: pitches $\{h_s\}_{s\in\mathbf{I}}$, lengths $\{d_s\}_{s\in\mathbf{I}}$ and accents $\{b_s\}_{s\in\mathbf{I}}$, $\{I = 1, ..., m\}$. Then let (V, d) be a metric space on a finite set of element (V). V and d depends upon the musical system we are considering.

Now, let's consider the linear graph G_l obtained by associating a vertex labelled by h_s to every element $h_s \in V$ and an oriented arrow $a_s : h_s \to h_{s+1}$ to every couple (h_s, h_{s+1}), so that $\partial_0 a = h_s$ and $\partial_1 a = h_{s+1}$.

We can then define a weight function $p : A_{G_l} \to \mathbf{Q} \times \mathbf{Q}$ by:

$$p : a_s \to (d_s, b_s) \in \mathbf{Q}^+ \times \mathbf{Q}^+ \;\; \forall s = 1, ..., m-1 \tag{9}$$

where $a_m : h_m \to h_1$ and $p(a_m) = (d_1, b_1)$.

Then we quotient the vertex set by identifying the vertex with the same label.

Definition 10. *Let M be a TF, we call* musical graph representing M *(and we write $G(M)$) the graph obtained by the process described above.*

Now we are going to analize the properties of the graph G(M) representing a TF M.

Proposition 6. *Musical graphs are eulerian, connected, oriented multigraphs.*

Proof. The proof is trivial if one considers the construction described above. In fact we have sent every interval of the original TF in an equally oriented arrow. The melody is a sequence of intervals, so it is clear that such a sequence represents an oriented trail in the graph which uses every edge once and once only. The closure of the trail is imposed by the definition, because we suppose the last interval being the last note-first note one.

Hamiltonian Graphs Serial music is an important part of contemporary music and also in diatonic and tonal contexts.

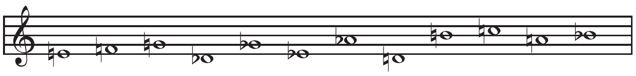
So an important class of TFs are those which correspond to series: they have an interesting interpretation in our model. In fact, series correspond to *hamiltonian graphs.*

The next proposition characterizes the TFs which contains a *series.*

Proposition 7. *A TF M contains a series iff its representative graph $G(M)$ is hamiltonian.*

Proof. In fact, given a series M the resulting graph is necessarily hamiltonian, because it contains all the intervals of the series. Viceversa an hamiltonian path in G represents obviously a series M.

Example 1. Let's consider a dodecaphonic series (cfr. [13])



Its representative graph is clearly C_{12}, which is hamiltonian.

Remark 5. Cyclic graphs C_n are a trivial example of series.

Remark 6. A particular case is represented by equilibrate series (cfr. [11]). Let's recall their definition.

Definition 11. *A TF is an* equilibrate series *iff its representative graph is cyclic (thus hamiltonian) and* $\forall a_i, a_j \in A_G, a_i \neq a_j$

$$d(\partial_0 a_i, \partial_1 a_i) \neq d(\partial_0 a_j, \partial_1 a_j) \qquad (10)$$

In other words equilibrate series present the greatest variety not only in vertex set but also in the arrows.

Equivalences One of the reasons why we needed to enlarge the concept of similarity was the recognition of new kinds of transformations; particularly we were interested in the permutation of melodic subfragments. These tranformations are important not only for musicologic research purposes in contemporary music but also for investigations in canonic-imitative music.

Now, let's analise two concepts of equivalence of TFs that will be central in the model.

Euler-Equivalence of TFs An equivalence concept which comes out from the representative graph construction is the one concerning the different trails in the graph.

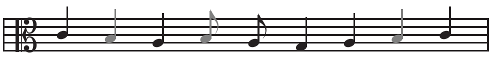
Definition 12. *We say that two TFs are* Euler-equivalent *iff they have he same representative graph.*

Let's try to better understand what this could mean from a musical point of view by some propositions.

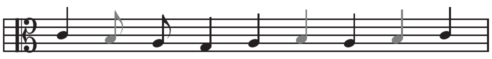
Proposition 8. *A graph is Eulerian iff it is decomposable in edge-disjoint cycles.*

Musically the proposition means that in an equivalence class there are TFs which admits a common cycle decomposition.

Example 2. Now, consider the TF (cfr. [19]):



and let's permute the cycles with the evidenced start and stop points (B,A,B) e (B,A,G,A,B):



The two TFs have exactly the same representative graph.

Therefore when we consider a particular musical graph we are really considering all the TFs corresponding to all the different eulerian circuits of the graph (with fixed starting point)

Musically this means that we *identify TFs obtained by particular permutations of subfragments*. We want to point out that these aren't arbitrary permutations. Otherwise there would be no advantages in respect of Tenney & Polansky's non ordered interval metrics ([1], [2]).

Is it possible to compute exactly the number of euler-equivalent TF by the next result.

Proposition 9. *Every class of euler-equivalent TFs has cardinality given by*

$$c \cdot \prod_{i=1}^{n} (d_i^+ - 1)! \tag{11}$$

where c is the number of equioriented spanning trees of the same representative graph.

Proof. The proposition follows from the Cayley theorem.

Equivalence of TFs Now we give a more general notion of equivalence, which includes also the standard tranformations of music theory.

Definition 13. *Two TFs are* equivalent *iff their representative graphs are isomorphic.*

Remark 7. Equivalence implies euler-equivalence.

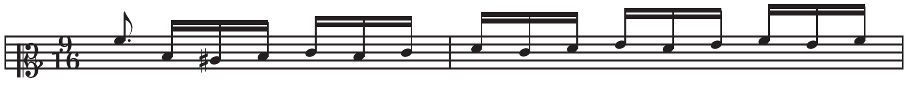
Remark 8. Standard melodic transformations are included into the isomorphism definition. In fact isomorphism implies an isometry between the metric spaces of vertices; therefore if we consider for example the standard equally tempered metric space (S^1) it is evident that tranformations like *transposition* and l'*inversion* are isometries $i : V \to V$.

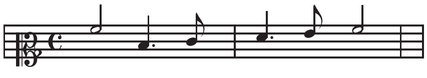
The *retrogradation* consists in the inversion of the orientation, so we have just to consider the opposite graph.

4.2 Subgraphs and Inclusions

Besides euler-equivalent TFs there is another interesting class of TF that can be obtained from a given one. The *eulerian subgraphs.* In fact it is possible to choice vertex and arc subsets such that the resulting graph remains eulerian.

Example 3. Consider the two TFs (cfr. [18])





Of course the second TF is an eulerian subgraph of the first one.

This is a quite particular case, which points out how we can recognise TF inclusions by a simple graph difference.

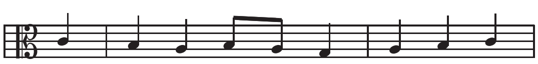
The necessary condition in this case is that the embellishments must be *closed circuits* of the graph.

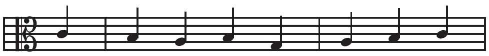
Thus we have the following proposition.

Proposition 10. *Let M be a TF and M' another TF obtained from M by the addition of closed embellishment to the notes of M. Then $G(M)$ is an (eulerian) subgraph of $G(M')$.*

Now we will define a notion of inclusion using the "weak inclusion" defined before.

Consider the musical theme (cfr. [19]):





It is clear, musically speaking, that the first TF is a variation of the second one.

Now we will try to formalize this concept by the next definition.

Definition 14. *We say that a TF A is included in a TF B if the representative graph of A is weakly included in the representative graph of B $G(A) \subseteq G(B)$ 7 such that $i : V_G(A) \rightarrow V_G(B)$ is an isometry.*

4.3 Necessary Conditions

The problem of deciding the inclusion of a TF into another one, using the "large" concept of inclusion, moved from the TFs to their representative graphs.

Teorically it may also be possible to use *string matching* tecniques, but there are at least three facts which lead to exclude those methods.

First of all, we should compare all the TFs undisctintively, and this would be particulary heavy from a computational point of view, expecially real-time.

Another handicap of the sequential approach is the fact that it's rigorously *note-dipendent*, i.e. it depends, in our graphic language, from vertex labelling. The transformation of TFs should be not contempled.

At the end, the natural equivalence class of TFs should be that concerning the isometries between the vertex set only, excluding permutations of sub-TFs.

Large part of our work consisted in searching for graph *invariants monotonic* respect the partial order relation of *inclusion* in the graph set defined in 7. Using graph invariants we could consider a TF with all its transformations by isometries. This process really does not depend on vertex labelling and so we can consider a TF toghether with all its transformed ones.

To fasten the inclusion test however it would be necessary to find out a set of good *necessary conditions*, which can reduct the set of TFs to compare. Now we will explore the condition so far identified.

Order and Size The first invariant we are going to consider are graph order and graph size.

Definition 15. *The* order *and the* size *of a TF M are the order $n(G)$ and the size $m(G)$ of the graph G representing M.*

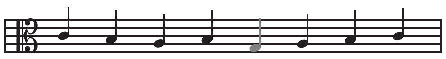
Of course we can observe that:

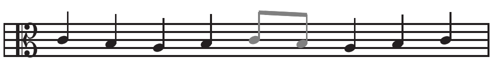
Remark 9. If $G \subseteq H$ then

$$n(G) \leq n(H) \qquad m(G) \leq m(H), \tag{12}$$

i.e. the number of vertex and the number of arrows in G have to be less than or equal to their respective in H.

From a musical point of view the condition concerning the arrows is evident (the fragment $M(H)$ must have more intervals than $M(G)$). Vertex condition is less trivial. In fact it says that the total number of distinct notes must increase or remain stationary in an inclusion, so we could never have a situation like this:



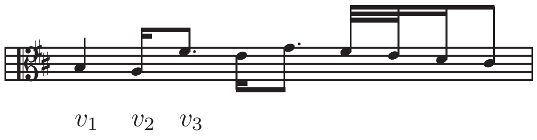


The second fragment includes G, but the first doesn't. These fragments aren't comparable. Moreover our inclusion relation induces a *partial* order on the musical graph set.

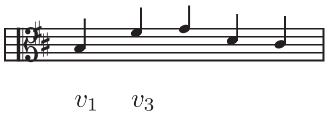
The Complexity Another significative invariant is the number of spanning trees of the graph representing the TF. At the graph level, an embellishment is obtained by the substitution of an arrow with an equioriented trail of the same total weight, with the same beginning and end vertex.

Consider the simplest variation: the insertion of one note.

Example 4. In this fragment



the first note of the *alla lombarda* rhythm, corresponding to vertex v_2 is the embellishment of the line



So, in the correspondent graph we have:

– first fragment: the arrows $a_1 = (v_1, v_2)$ and $a_2 = (v_2, v_3)$
– second fragment: the arrow $b_1 = (v_1, v_3)$

where $|b_1| = |a_1| + |a_2|$.

In this example, vertex v_2 has minimum degree ($v^+ = v^- = 1$), i.e. there are no other trails passing trough it, because we wanted to point out the non decreasing property of graph complexity in the substitution of M_2 with M_1.

In fact the second fragment (M_2) has $k(G(M_2))$ spanning tree and if we replace the arrow b_1 with the trail a_1a_2 such trees continue to span the graph (M_1).

This fact pointed out by the example helds also with added vertex belonging to the first graph. Let's better formalize this fact with the following definition.

Definition 16. *The* complexity $k(M)$ *of a TF* M *is the complexity of its representative graph* $g(M)$ *(cfr. def. 9).*

We would remind that

Remark 10. The complexity of a graph is equal to the number of spanning trees of the graph

Proposition 11. *Let* M *be a TF and* M' *a variation of* M *obtained by the insertion of notes. So we have* $k(M) \leq k(M')$.

Proof. At the representative-graph level, the process which transforms M in M' consists of a substitution of oriented arrows by oriented trails. In this way the proposition follows from the 3 and 4.

The Degree Sequence Of course the inclusion of a TF M in another TF M' implies that the number of corresponding notes can only increase.

Definition 17. *The degree sequence of a TF M is the valence sequence of its representative graph $G(M)$.*

Proposition 12. *If a TF M is included into another TF M' then the respective valence sequences are such that*

$$d_i(M) \leq d_i(M') \ , \forall i \in I \tag{13}$$

Proof. The proposition follows by the definition of inclusion of TFs.

4.4 The Metric on V_G

By definition, given a musical graph G, the set V_G is a metric space.

It's really important considering the note set only as a metric space as we are interested in invariant objects of all musical transformations. The lonely really important facts are the distance ratios between the notes.

Certainly, the most general possible transformation in our contest is an *arbitrary permutation* of nodes; however this may cause an outgoing of the isometry set. Then, even a learned listener should run into serious difficulties in recognising such a transformation. Similarly, if wethink of changing the rhythm, the probabilities of recognise the TF should tend to zero.

We would remind at this point the musical transformation that a TF can undergo:

1. transposition;
2. specular inversions;
3. retrogradations.

Thus it is possible to give a necessary condition for the inclusion that will have a great relevance in recognising those transformations:

Proposition 13. *If a TF M is contained into another TF M' then the inclusion function $i_{V_G(M)}$*

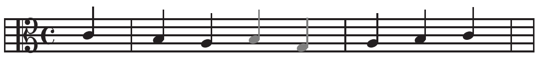
$$\begin{tikzcd}
A_G(M) \arrow[r, tail, "i_{A_G}"] \arrow[d, tail, "\partial_0, \partial_1"'] & \mathcal{P}(A_G(M')) \arrow[d, tail, "\partial'_0, \partial'_1"] \\
V_G(M) \arrow[r, tail, "i_{V_G}"'] & V_G(M')
\end{tikzcd} \tag{14}$$

is an isometry.

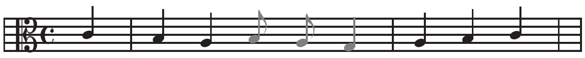
Proof. The proposition is obvious using the definition of (weak) inclusion and the definition of musical tranformations.

4.5 The Power Graph

Let's consider the TFs (cfr. [19])



v_1 v_3



v_1 v_2 v_3

They differ by a passing note (closure of the third). At the graph level we can observe that the arrow $a_1 : v_1 \to v_3$, wich is present in the first graph, is replaced by two arrows $a_1 : v_1 \to v_2$ and $a_2 : v_2 \to v_3$ in the second one, so a trail of lenght 2 replaced an arrow.

Let's remember the *transitive closure* operation.

Definition 18. *Let G be a graph. We call* transitive closure *of G the graph $\overline{G}$ such that $V_{\overline{G}} = V_G$ and $A_{\overline{G}}$ contains all and only the arrows such that:*

$$a_i = b_i b_j \quad \wedge \quad \partial_1 b_i = \partial_0 b_j \ , \ b_i, b_j \in A_G \tag{15}$$

Remark 11. The transitive closure is an internal unary operation on the set of eulerian graphs; in fact we can observe that:

Proposition 14. *If G is an eulerian graph of size m, then $\overline{G}$ is also eulerian of size $2m$.*

Proof. Given an eulerian cycle in G, the arrows of $\overline{G}$ are the arrows of G plus a number of arrows equal to the number of couple of adjacent arrows of G. Thus is obvious that $\left|\overline{G}\right| = 2m$. By construction, the graph resulting from a closure operation on an eulerian graph is obviously eulerian.

From a musical point of view the operation consists of an union of two TFs: $M(G)$ and the TF obtained from $M(G)$ taking all the notes of $M(G)$ but proceeding by jumps.

Iterating the process we can obtain the *k-th powers* of a graph.

Definition 19. *We call k-th power M^k of a TF $M(G)$ the TF whose representative graph is obtained from the graph $G(M)$ iterating (k-1) times the transitive closure operation.*

The interesting result which comes out from those definitions is the next proposition.

Proposition 15. *Let M and M' be two TFs. M' contains a variation of M by adding single notes if and only if exists an isometry $i : V_{G(M)} \hookrightarrow V_{H(M')}$ such that the equation*

$$G \setminus H \setminus (H \setminus G)^2 = \emptyset \tag{16}$$

is satisfied.

Proof. If M' contains a variation of M which is obtained by adding no more than one note between two consecutive notes of M so we will therefore have a graph inclusion $i : G(M) \hookrightarrow H(M')$ such that the set A_H will be partitionable in classes formen by trails η such that:

$$\eta = i_A(a) = h,\ h \in A_H \tag{17}$$

or

$$\eta = i_A(a) = h_1 h_2\ ,\ \ \mathit{with}\ \partial_1 h_1 = \partial_0 h_2\ \ h_i \in (A_H \setminus A_{i(G)}) \tag{18}$$

In the first case we have $\eta \in A_H \cap A_{i(G)}$ though in the second case $\eta \in A_{(H\setminus i(G))^2}$. So the 16 is satisfied.

Vice versa, if exists an isometry $i : V_{G(M)} \hookrightarrow V_{H(M')}$ such that it satisfies the 16, we'll have $G \setminus H \subseteq (H \setminus G)^2$; hence $\forall a \in i(A_G)$ will be:

$$a \in A_H \cap A_{i(G)} \quad \vee \quad a = h_1 h_2\ ,\ \ \mathit{with}\ \partial_1 h_1 = \partial_0 h_2\ \ h_i \in (A_H \setminus A_{i(G)}) \tag{19}$$

Hence we can partition A_H following the definition.

Finally we've obtained an operative sufficient and necessary condition for the inclusion which can be usefully implemented into an algorithm for the recognition of the inclusions.

4.6 Graph Metric

Now we can define a metric in the set of musical graphs.

Definition 20. *Given two musical graphs G ed H we define the distance between G and H as the number:*

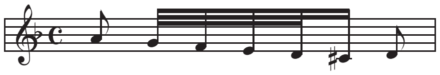
$$d(G, H) = \frac{|A_G \ominus A_H|}{|A_G \sqcup A_H|} \tag{20}$$

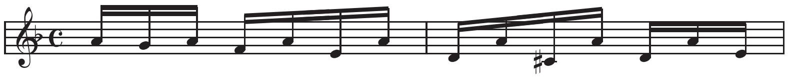
where $\ominus$ is the symmetric difference *operator.*

Remark 12. Naturally the function defined above satisfies the metric axioms (inherited by the insiemistic metric), so the set of musical graphs together with the distance function d is a *metric space.*

From a musical point of view, the (normalized) metric defined above is not satisfactory as it assignes very high distances to the couples of TF which are similar (for example a couple of fragments which differs only for passing tones).

Example 5. Let's consider the two TF (cfr. [20], BWV 565)





The second TF is clearly obtained inserting the tone A between each couple of consecutive tones of the first TF. Yet the distance calculated by the 20 is:

$$d(G,H) = \frac{|A_G \ominus A_H|}{|A_G \sqcup A_H|} = \frac{5+13}{20} = \frac{9}{10}. \tag{21}$$

Absolutely too high!

The problem is that such a distance do not consider the lenght-two graph trails at all. Musically, this means to ignore, for example, *passing tones*, *neighbor tones*, etc..

Indeed we need a more general notion of similarity which considers and, if possible, gives a different weight to these musical facts.

4.7 Similarity Function

Now let's give the notion of similarity function between graphs which will be useful to estimate the similarity between two TFs.

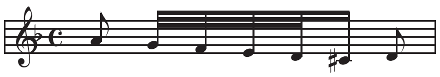
Definition 21. *Let M and M', $M \leq M'$, be two TFs with representative graphs $G = G(M)$ and $H = H(M)$. Then, given $r \in \mathbf{N}$, we call* r similarity function ordine r between M and M' *the function:*

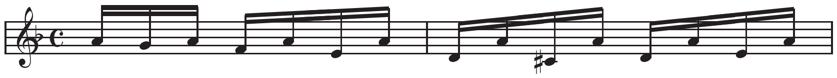
$$\sigma(M,M') = \sigma(G,H) = \max_{\phi} \sum_{i=1}^{r} \alpha_i \frac{|G| - \left|G \setminus H^i \setminus H^{i-1}\right|}{|G|} \tag{22}$$

where $H^0 = \emptyset$, α_i are positive coefficients which depend upon the weight assigned to the different trail lenghts and ϕ varies among all the possible isometries from V_G to V_H.

Remark 13. The function $1/\sigma$ isn't a metric on the set of TFs, because $\sigma(M,M') \neq \sigma(M',M)$.

Example 6. Let's consider the TFs A e B below (cfr. [20]):

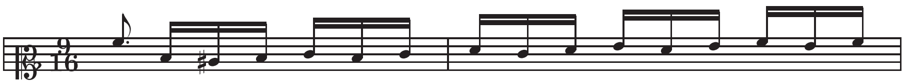


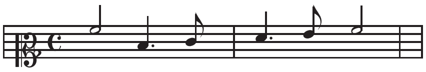


The first TF (A) is clearly contained in the second (B) and the function which realizes the max is the identity function. Let's calculate the similarity function with $r = 1$ and $\alpha_1 = \alpha_2 = 1$. We have:

$$\sigma(A, B) = \frac{1}{7} + \frac{6}{7} = 1 \tag{23}$$

Example 7. Now, consider the two fragments (cfr. [18])





The second one, as we've already pointed out in 4.2, is even an eulerian subgraph of the first one because its intervals are present also in the first one.

Let's calculate the similarity function with $r = 1$ and $\alpha_1 = \frac{1}{2}$, $\alpha_2 = 12$. We have:

$$\sigma(A, B) = \frac{1}{2} + \frac{1}{2} = 1 \tag{24}$$

5 Algorithmic Implementation

We tested the algorhitms deriving from the theory by a software implementation of the model. The software prototype has essentially the function of automatize the process of comparison of TFs.

As we've described in analysing the model, there are various necessary conditions for the inclusion of TFs.

The inclusion-monotone invariants constitute an integrant part of the software and are necessary in order to delete any comparison which could surely fail.

Consequently, the algorithm divides into three parts:

1. construction of the representative graphs of TFs;
2. test of necessary conditions for the inclusion;
3. calculation of the similarity function.

Graphs have been implemented by a java class `Grafo` endowed with methods and data structures to permit all operations.

6 Summary and Conclusion

The model presented here enlarge the similarity class of tematic fragments in respect of the other models known in literature.

Particularly, it is clear that the more a TF presents variety in melodic and interval construction the smaller becomes its euler-equivalence class. Viceversa, TFs with repetitions tends to be more similar, increasing the cardinality of their eulerian class. So the model is coherent with the common musical intuition.

7 Future Work

The graphical approach can be developed towards numerous directions. First of all we have to increase the number and the power of necessary conditions which are musically significant. At the moment, we are studing the relevance of particular polynomials defined by graphs.

Afterwards the rhythm and accentual dimension, defined by arrows weights, have to be integrated into the model. The resistive metrics seem to give good results for the rhythm.

Another developement line is the optimization of the algorhitms and their software implementation.

8 Acknowledgments

The authors wish to acknowledge the partial support of this project by Italian MIUR (FIRB "Web-Minds" project N. RBNE01WEJT_005) and the Italian National Research Council, in the framework of the research program "Methodologies, techniques, and computer tools for the preservation, the structural organization, and the intelligent query of musical audio archives stored on heterogeneous magnetic media", Finalized Project "Cultural Heritage", Subproject 3, Topic 3.2, Subtopic 3.2.2, Target 3.2.1. We also want to acknowledge the members of the MAX WG for their cooperation and interest in our work.

This work has been made possible by the efforts of researchers and graduate students of LIM.

References

1. James Tenney e Larry Polansky. *Temporal gestalt perception of music: a metric space model.* Journal of Music Theory, 24, 205-241, 1980.
2. Larry Polansky. *Morphological metrics.* Journal of New Music Research, 25, 289-368, 1996.
3. Larry Polansky. *More on morphological mutation functions: Recent techniques and developements.* Proceedings of the International Computer Music Conference, 50-60, 1992.
4. Fred Lerdahl e Ray Jackendoff. *A Generative Theory of Tonal Music.* MIT Press, 1996.
5. Eleanor Selfridge-Field. *Melodic Similarity.* MIT Press, 2000.
6. Alberto Pinto. *Modelli formali per misure di similarità musicale.* Tesi di Laurea in Matematica, Università degli Studi di Milano, a.a. 2002-2003.
7. Frank Harary. *Graph Theory.* Addison-Wesley, 1969
8. Béla Bollobás. *Modern Graph Theory.* Springer, 1998.
9. Chris Godsil, Gordon Royle. *Algebraic Graph Theory.* Springer, 2001.
10. Fred Buckley, Frank Harary. *Distance in Graphs.* Addison-Wesley, 1990.
11. Enzo Martissa. *Analisi e sintesi di processi pseudo-musicali.* Tesi di Laurea in Fisica, Università degli Studi di Milano, a.a. 1976-1977.
12. Luigi Verdi. *Organizzazione delle altezze nello spazio temperato.* Ensemble '900, 1998.
13. Arnold Schönberg *Harmonielehre.* Leipzig-Wien, 1911.
14. M.Baroni, R.Dalmonte, C.Jacobini. *Le regole della musica.* EDT, 1999.
15. Goffredo Haus. *Elementi di informatica musicale.* Jackson, 1984.
16. Curtis Roads. *The computer music tutorial.* MIT Press, 1996.
17. Guido Farina. *Manuale di armonia.* Carish, 1946.
18. Johann Sebastian Bach. *Die Kunst der Fuge.* BWV 1080, ed.Peters, 1967.
19. Johann Sebastian Bach. *Einige canonische Veränderungen über das Weinachtslied: 'Vom Himmel hoch, da komm'ich her'.* BWV 769, ed.Peters, 1967.
20. Johann Sebastian Bach. *Orgelwerke.* ed.Peters, 1967.

A Content-Based Music Retrieval System Using Representative Melody Index from Music Databases

Jae-Yong Won, Jae-Heon Lee, KyongI Ku, Jaehyun Park, Yoo-Sung Kim

School of Information and Communication Engineering,
Inha University, Incheon 402-751, Korea
yskim@inha.ac.kr

Abstract. For content-based music retrieval, since not only the correctness of retrieval results but also the performance of retrievals is important, there are great needs for efficient content-based music retrieval systems that can quickly retrieve the relevant music on demand from large music database with low storage overhead. In this paper, we describe the design and implementation of a content-based music retrieval system in which the representative melody index is systemically constructed and used to support quick and appropriate retrievals to users' melody queries. By using the proposed system for digital music libraries, it could save up to 65% of index space than that for the whole motifs index while the appropriateness of retrieval results is maintained.

1 Introduction

There have been strong needs for investigation and development of content-based multimedia information retrieval systems in order to support the effective and efficient retrieval for multimedia information. While, for image or video information, several content-based retrieval systems have been developed, for music information, most of current working music retrieval systems are based on only metadata of music, such as title, composer, singer, and words of song([1], [2], [3], [4]). However, these traditional systems based on metadata have the major restriction that users should recall and specify metadata of music they want as users' queries. This restriction is unnatural due to the general fact that people prefer to remember a part of music itself rather than its metadata. Therefore, content-based music retrieval system in which some melodies are used as queries in order to retrieve music information is essentially required.

As content-based music retrieval systems, relatively few systems have been developed([5], [6], [7], [8]). In these systems, users specify query melodies by humming, by playing, or by drawing a part of music remembered as the representative melodies. Here, by the representative melodies of a music we mean that they are semantic delegation of music's melodies such as the first melody, the climax melody, and the repeated theme melodies of the music and people can remember these melodies for the music so that users are likely to use these melodies as query melodies([9]). Then, the system retrieves the music information according to the similarity between user's query melody and the melodies of the underlying music database.

However, these previous content-based music retrieval systems do not have the effective indexing mechanism that is helpful to improve the appropriateness and performance of retrievals. Hence, in traditional content-based music retrieval systems, users may face with long response time, since those systems need time-consuming syntactic processing for retrievals in which an approximate matching between query melody and all melodies of underlying music database should be performed.

Some of the previous systems([8], [10]) have the first melody index in which only the beginning part of each music file is included. In general, however, since people are likely to remember music by its representative melodies not only by just its first melody users can input other representative melodies not the first melody as query melodies. Then, they could not get the music information they want. In other words, these systems with the first melody index cannot support user queries of all possible representative melodies including climax melody and repeated theme melodies of music.

To remedy the above problem, some researches([11], [12], [13], [14]) have concentrated on extracting repeated theme melodies from a music file. In [11] and [12], since authors consider the exactly repeated patterns in a music file as theme melody, a melody exactly repeated two or more times extracted as a theme melody from the music. In general, however, according to the property of theme melody in musicology, *theme reinstatement*, a theme melody should be repeated more than once with some variances within a music file([15]). In other words, the theme melody index in which only the exactly repeated theme melodies extracted by the mechanism of [11] or [12] is not enough to support users' queries of the representative melodies.

However, in our previous work([14]), we proposed a theme melody extraction mechanism in which an extended graphical clustering algorithm is used for grouping the approximately repeated melodies into a cluster with considering musical composition forms and a melody is extracted from each cluster as an approximately repeated theme melody. In addition to the extraction of the approximately repeated theme melodies from a music file, the first melody and the climax melody of the music are augmented into the final representative melody set. Thus, the representative melody index can well support users' queries in which these kinds of melodies are used.

In this paper, we discuss the design and implementation of a content-based music retrieval system in which the representative melody index is used to support effectively and efficiently users' queries in which some part of the first melody and/or the climax melody and/or the approximately repeated theme melodies of music files they want are used. We also discuss the retrieval and the relevance feedback procedures using the representative melody index for several types of users' queries.

The rest of this paper is organized as follows. Section 2 discusses the indispensable features for content-based music retrieval system and the overall architecture of the proposed system to support these indispensable features. In section 3, we discuss the systematic construction of the representative melody index when a music file is registered into music database. In section 4, we discuss the procedures of content-based music retrieval using the representative melody index. We also discuss the performance of the system. Finally, this paper is concluded with future works in section 5.

2 Architecture of Content-Based Music Retrieval System

In this section, we discuss the ideal architecture for content-based music retrieval systems. For it, first we summarize the essential requirements of content-based music retrieval systems. Then, according to the requirements we introduce the overall architecture of the content-based music retrieval system developed in this work.

2.1 Requirements for Content-Based Music Retrieval Systems

In content-based music retrieval systems, as user's query, since humming, playing or drawing a part of music can be used, content-based music retrieval systems should have a user interface that is able to support these types of querying. The interface transforms user's query into an intermediate representation in order to compute the similarity between query melody and melodies in music database. In fact, however, in order to enhance the response times of retrievals from huge music database, comparing the intermediate representation of query melody to all of melodies in the huge music database must be away from. For this purpose, content-based music retrieval systems should have the index that has the representative melodies of the music database in order to compare user's query first instead of the huge music database. For the representative melody index, it is needless to say that during the hit ratio of the index is maintained appropriately, the smaller index is, the better response time users get. Here, the hit ratio means the rate of the total number of users' queries to the number of the matched queries successfully in the representative melody index. To keep the index small, only the appropriate representative melodies that are likely to be used as users' query melodies should be extracted from music database.

Since user's query is not so accurate, the system should compute the similarity between the query melody and the representative melodies of music database. Also, the ranks of the retrieval results are based on the similarity values to the query melody. Furthermore, since visual and/or auditory user interfaces are helpful for users to validate whether the retrieval results are appropriate to users' queries, people may prefer a content-based music retrieval system that has visual and auditory user interfaces.

When users do not satisfy the retrieval results, users submit either a totally new query or a modified query from the previous query or previous results. In information retrieval community, this kind of work for modifying the original query to enhance gradually the appropriateness of retrieval results is referred to as *user relevance feedback*. Hence, to allow users to get more correct results from the previous query, content-based music retrieval systems should have a user relevance feedback.

2.2 Overall Architecture to Satisfy the Requirements

The overall architecture of the content-based music retrieval system developed in this work is shown in Fig 1. It consists of User interface(Registration interface and Query & Result interface), RM extractor & indexer, M-tree engine, Music database, MIDI generator, Query processor, and Ranker.

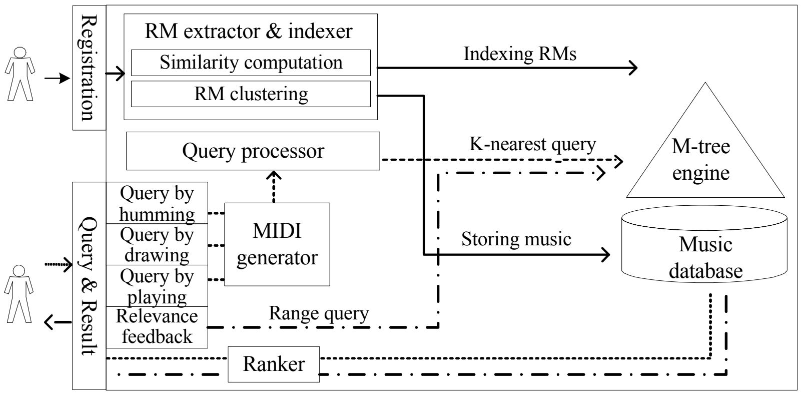


Fig. 1. Architecture of the Proposed Content-based Music Retrieval System

User interface. There are two types of user interfaces; Registration interface and Query & Result interface. Query & Result Interface consists of three modules for querying(query by humming, query by drawing, and query by playing) and two modules for retrieval results(one is for viewing the music score and the other is for listening the music). Up to now, since the drawing interface is well tested and is good for viewing the content of users' queries, we will use this interface hereafter. To improve the satisfaction degree for the retrieval results, users can use the previous query or the previous retrieval results to make a next promotional query. For this purpose, user interface should include also the user relevance feedback module.

RM extractor & indexer. From the new music file submitted via Registration interface, the representative melodies are extracted and included into the representative melody index by RM extractor & indexer. This work is accomplished by the unit of motif since it is the minimum meaningful unit in music semantics([15]). This module has two primary sub-modules; Similarity computation and RM clustering. Similarity computation module computes the similarity values of all pairs of motifs of the music file and finally constructs a similarity matrix of the music file. And, RM clustering module classifies motifs of a music file into one or more groups in each of which only the similar motifs to each other are included. The more details of extracting and indexing representative melodies are discussed in section 3.

M-tree engine. The representative melody index is implemented by M-tree, a multidimensional indexing scheme. To place the extracted representative melodies into the metric space of M-tree, we choose the average length variation and the average pitch variation of representative melody as the key features. Also, the extracted representative melodies are stored with their signatures. The signatures of representative melodies are used to filter out and to rank retrieval results.

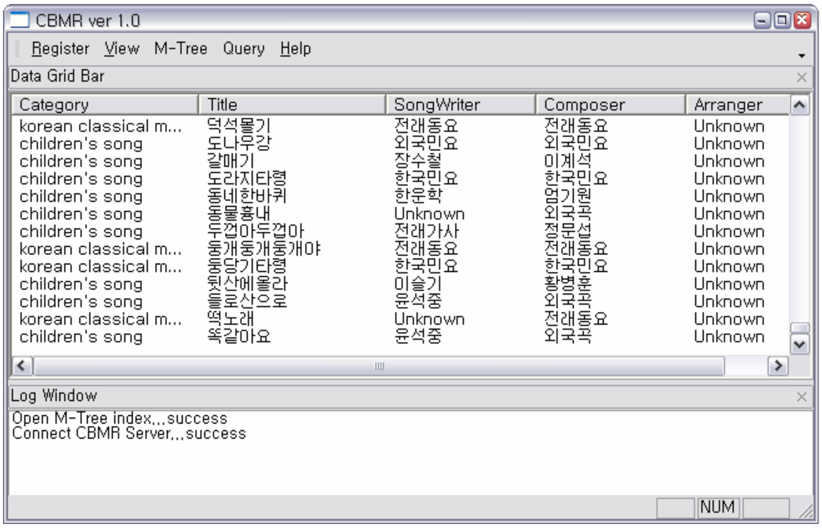
Music database. The music files are stored in music database with metadata(time signature, key signature, composer, singer etc.), file location, and other housekeeping information. The metadata is for text-based music retrieval and the file location is used to return the corresponding music file as retrieval result.

MIDI generator. In this work, we assume that music file is in MID(musical instrument digital interface) format since it is a well-known standard for computer music. MIDI generator transforms a query melody of humming, drawing, or playing into MIDI format as the intermediate format.

Query processor. Query processor first makes the retrieval features and the signature of user query melody. It retrieves the relevant melodies from the representative melody index of M-tree by using *k*-nearest neighbor search. The more details concerned on retrieval and user relevance feedback are discussed in section 4.

Ranker. Ranker gets rid of false-dropped results from the retrieval results and decides ranks of the retrieval results according to the distances between the query's signature and the signatures of retrieval results. Since a query melody is of different length from representative melodies in the representative melody index, we use time-warping distance function([19]) to compute the distance between them.

A prototype of the content-based music retrieval system using representative melody index is developed by using Microsoft's Visual C++ 6.0 and is working as a desktop application on Microsoft's Windows systems. The first screen of the developed system is shown in Fig. 2. The main window is for listing up the music files already registered and the lower window is for system logging.



Category	Title	SongWriter	Composer	Arranger
korean classical m...	덕석몰기	전래동요	전래동요	Unknown
children's song	도나우강	외국민요	외국민요	Unknown
children's song	갈매기	장수철	이계석	Unknown
children's song	도라지타령	한국민요	한국민요	Unknown
children's song	동네한바퀴	한운학	엄기원	Unknown
children's song	동물흉내	Unknown	외국곡	Unknown
children's song	두껍아두껍아	전래가사	정문섭	Unknown
korean classical m...	둥개둥개둥개야	전래동요	전래동요	Unknown
korean classical m...	둥당기타령	한국민요	한국민요	Unknown
children's song	뒷산에올라	이슬기	황병훈	Unknown
children's song	들로산으로	윤석중	외국곡	Unknown
korean classical m...	떡노래	Unknown	전래동요	Unknown
children's song	똑같아요	윤석중	외국곡	Unknown

```
Open M-Tree index...success
Connect CBMR Server...success
```

Fig. 2. The First Screen of the Content-based Music Retrieval System

3 Systematic Construction of Representative Melody Index

3.1 Extraction of Representative Melodies

The summarized procedure for the extraction of the representative melodies(RMs) from a music file is shown in Fig. 3.

When a music file is submitted, the features such as time-signature, pitch and length of notes are extracted from the submitted music file. By using these features, we decompose a music file into the set of motifs, since a motif is the semantic unit of music composition. Then we compute the similarity values between all pairs of the motifs by using the similarity computation algorithm([16]). Then, the similarity matrix can be constructed. The motifs of a music file are clustered based on the similarity values by using the proposed RM clustering algorithm that considers the musical composition forms. The detail of the clustering algorithm is in [14].

From each clusters, we extract an approximately repeated theme melody as a representative melody based on the position or role of the motif within the music. If a cluster includes the first motif or the climax motif, we extract that motif as the representative melody from the corresponding cluster. Otherwise, we extract a RM from each cluster to allow the extracted melody to be the center position of the cluster in metric space of M-tree([17]). After extraction of RM from each cluster, if the first motif or the climax motif of a music file does not exist in the extracted melody set, we add them to the final set of representative melodies for the music file.

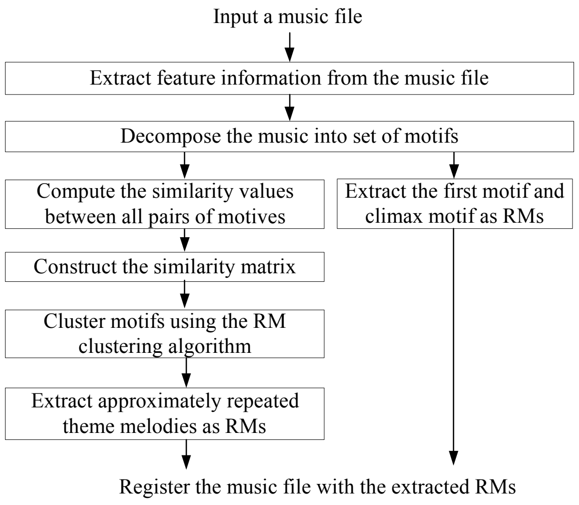

Input a music file
Extract feature information from the music file
Decompose the music into set of motifs
Compute the similarity values between all pairs of motives
Extract the first motif and climax motif as RMs
Construct the similarity matrix
Cluster motifs using the RM clustering algorithm
Extract approximately repeated theme melodies as RMs
Register the music file with the extracted RMs

Fig. 3. Procedure for Extracting Representative Melodies from Music File

As an example of extracting the representative melodies, we consider a Korean children song, 'Spring of Hometown' of Fig. 4. The song consists of 8 motifs and has a pattern as its musical composition form, *A*-*B*-*C*-*D*-*E*-*B'*-*C*-*D*. From the musical composition form, we can expect that 8 motifs should be clustered into three similar groups, {*B*, *B'*}, {*C*, *C*}, and {*D*, *D*}.

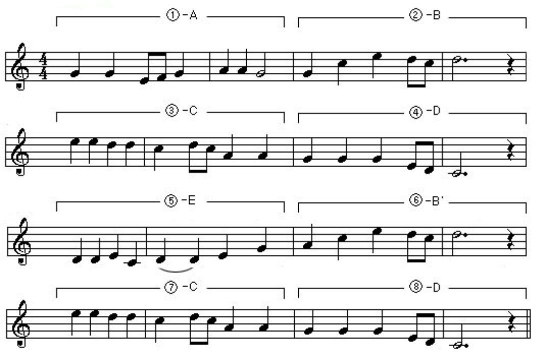
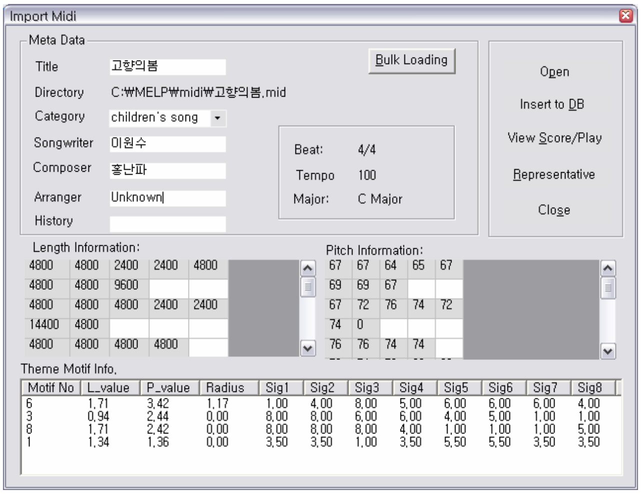


Fig. 4. The Music Score of a Korean Children Song ‘Spring of Hometown’

When we input the MIDI of ‘Spring of Hometown’ in Fig. 4 for registration by choosing the ‘Register’ menu in Fig. 2, we can see the screen of Fig. 5. We can see the information for representative melodies extracted from the music at the lower window.



Meta Data

Title	고향의봄
Directory	C:₩MELP₩midi₩고향의봄.mid
Category	children's song
Songwriter	이원수
Composer	홍난파
Arranger	Unknown
History	

Beat:	4/4
Tempo	100
Major:	C Major

Length Information:

4800	4800	2400	2400	4800
4800	4800	9600		
4800	4800	4800	2400	2400
14400	4800			
4800	4800	4800	4800	

Pitch Information:

67	67	64	65	67
69	69	67		
67	72	76	74	72
74	0			
76	76	74	74	

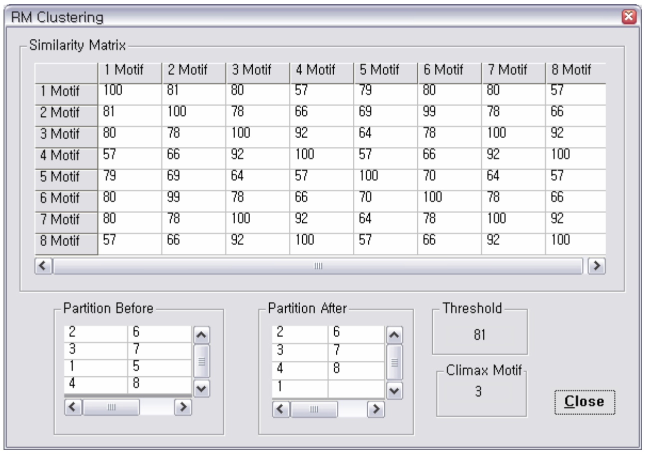
Theme Motif Info.

Motif No	L_value	P_value	Radius	Sig1	Sig2	Sig3	Sig4	Sig5	Sig6	Sig7	Sig8
6	1,71	3,42	1,17	1,00	4,00	8,00	5,00	6,00	6,00	6,00	4,00
3	0,94	2,44	0,00	8,00	8,00	6,00	6,00	4,00	5,00	1,00	1,00
8	1,71	2,42	0,00	8,00	8,00	8,00	4,00	1,00	1,00	1,00	5,00
1	1,34	1,36	0,00	3,50	3,50	1,00	3,50	5,50	5,50	3,50	3,50

Fig. 5. Registration Window for ‘Spring of Hometown’

To see the more details of RM clustering, when users click the ‘Representative’ button of the rightmost frame we can see the screen of Fig. 6. In this screen, we can see the similarity matrix, the clustering results with threshold value by the RM algorithm of [14], and the climax motif of the music file. From the similarity matrix, we can easily recognize that ‘Spring of Hometown’ has the musical composition form of *A*-

B-C-D-E-B'-C-D since the similarity values between 3rd and 7th motifs, 4th and 8th motifs are 100 and the similarity value between 2nd and 6th is almost near to 100, exactly 99. That means we have final three clusters {2, 6}, {3, 7}, {4, 8}. From these clusters, as shown in Fig. 5, we extract the motifs 6, 3, 8 as the representative melodies, respectively. And 1st motif is augmented into the final set as the first melody. Note that 3rd motif is also the climax melody of the music file.



RM Clustering

Similarity Matrix

	1 Motif	2 Motif	3 Motif	4 Motif	5 Motif	6 Motif	7 Motif	8 Motif
1 Motif	100	81	80	57	79	80	80	57
2 Motif	81	100	78	66	69	99	78	66
3 Motif	80	78	100	92	64	78	100	92
4 Motif	57	66	92	100	57	66	92	100
5 Motif	79	69	64	57	100	70	64	57
6 Motif	80	99	78	66	70	100	78	66
7 Motif	80	78	100	92	64	78	100	92
8 Motif	57	66	92	100	57	66	92	100

Partition Before

2	6
3	7
1	5
4	8

Partition After

2	6
3	7
4	8
1	

Threshold: 81

Climax Motif: 3

Fig. 6. RM Clustering Result for 'Spring of Hometown'

3.2 Construction of Representative Melody Index

As discussed in section 2.2, to place melodies into the metric space of M-tree, we compute the average length variation and the average pitch variation of each melody and the radius of each cluster. If we assume that a representative melody of *n/m* times signature has *k* continuous notes, $[(l_1, p_1), (l_2, p_2), \ldots, (l_k, p_k)]$, where l_i and p_i are the length and pitch of *i*-th musical note in the melody, respectively. The average length variation $\bar{l}$ and the average pitch variation $\bar{p}$ are computed by Equation (1) and (2), respectively. In Equation (1), the first term denotes the average length difference of *k* musical notes in the representative melody to the dominator *m* of the times of the music and the second term denotes the average value of *k-1* length differences between continuous *k* musical notes. Similarly, in Equation (2), the first term denotes the average value of pitch differences between the first musical notes and the following *k-1* ones and the second term is for the average value of *k-1* pitch differences between *k* continuous musical notes. And the distance *d(v, u)* between two theme melodies $u(\bar{l}_u, \bar{p}_u)$ and $v(\bar{l}_v, \bar{p}_v)$ is computed by the Euclidean distance in 2-dimensional space. The radius of a cluster stands for the maximum distance between the extracted representative melody of a cluster and other melodies in the cluster in 2-dimenstional metric space of M-tree.

$$\bar{l} = ((\sum_{i=1}^{k} | m - l_i |) / k + (\sum_{i=1}^{k-1} | l_{i+1} - l_i |) / (k-1)) / 2 \qquad (1)$$

$$\bar{p} = (\sum_{i=1}^{k-1} (\frac{| p_1 - p_{i+1} | + | p_{i+1} - p_i |}{2})) / (k-1) \qquad (2)$$

At the lower window of Fig. 5, we can see the average length variations and the average pitch variations of the representative melodies and the radiuses of their clusters, respectively. Also, if we click 'M-tree' menu of the main screen of the developed system shown in Fig. 2, we can see the logical view of the representative melody index that is picture as a 2-dimensional graph as shown in Fig. 7.

However, if we use only the average length variation and the average pitch variation of a melody to place it in 2-dimensional metric space of the representative melody index, there might be two melodies that have same $\bar{l}$ and $\bar{p}$ values even if they are mainly different in melody patterns to each other. To distinguish these melodies correctly and finally to return only the appropriate retrieval results for users queries, we use the melody signatures each of which represents the variation pattern of a melody in more details according to the elapsed time. Hence, a signature of a melody is denoted as a time-series data of $<s_1, s_2, s_3, s_4, ..., s_t>$. In this work, we convert a melody of one motif length into a time-series data of 8 sequences. That is, a representative motif is translated into a time-series data of $<s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8>$ according to the conversion scheme proposed in our previous work([18]).

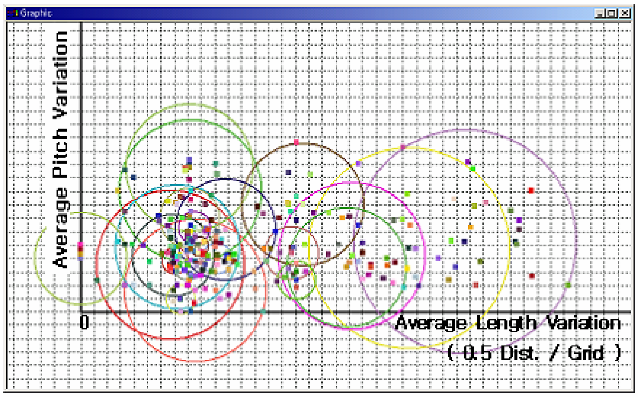


Fig. 7. Logical View of 2-Dimensional Metric Space of the Representative Melody Index

At the lower window of Fig. 5, we can see also the signatures of the extracted representative melodies, respectively, each of which is of 8 sequences. As discussed in section 2.2, the signatures of the melodies in the representative melody index is used to get rid of the false drop from retrieval result in addition to ranking the retrieval results. More detailed description of using signatures for filtering out and ranking is in section 4.1.

4 Content-Based Music Retrieval from Melody Query

4.1 Procedure of Content-Based Music Retrieval

The summarized procedure for the content-based music retrieval using the representative melody index is in Fig. 8. When a user's melody query with the expected number of results *r* is submitted to the query interface, the system extracts the feature information form the query and composes the average length variation, the average pitch variation, and the signature of the query melody in similar way for registering new music file. With these features of the query, we do *k*-nearest neighbor searching from M-tree of the representative melody index. In this work, we use *2r* as *k* value in *k*-nearest neighbor searching in order to enrich the candidates for more appropriate retrieval results.

To filter out inappropriate melodies from *2r* candidates and to rank the retrieval results, we compute the similarities between the query's signature and those of *2r* candidates by time-warping distance function. As well known, time-warping is able to compute the distance between two melodies of different lengths. Hence, we can retrieve the appropriate melodies from the representative melody index even thought users submit query melodies that are not of one motif length. According to the similarity values, we choose top *r* melodies from *2r* candidates and decide the ranks of the final retrieval results.

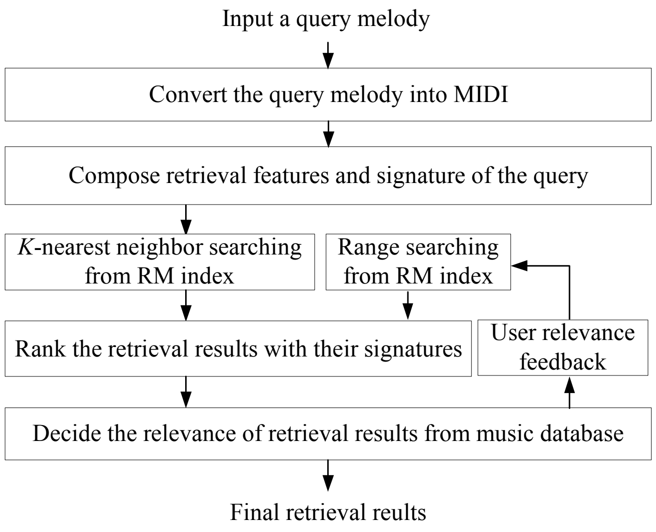
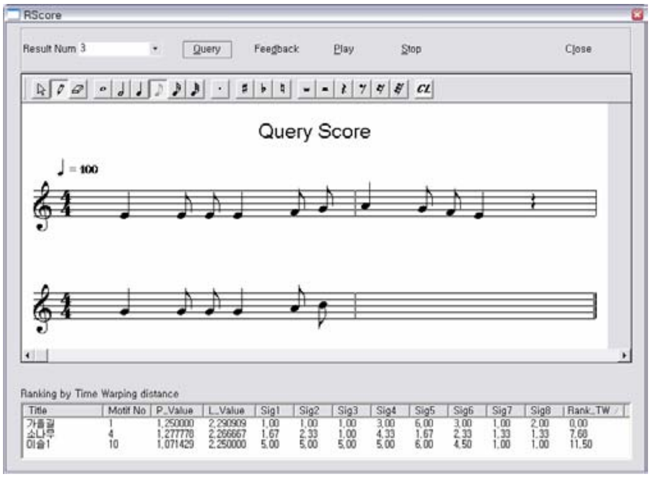

Input a query melody
Convert the query melody into MIDI
Compose retrieval features and signature of the query
K-nearest neighbor searching from RM index
Range searching from RM index
Rank the retrieval results with their signatures
User relevance feedback
Decide the relevance of retrieval results from music database
Final retrieval reults

Fig. 8. Procedure of Content-based Music Retrieval Using Representative Melody Index

Users view the final retrieval results based on their ranks and check the relevance for each music file in the retrieval results via viewing its music score with a visual interface and/or listening the music with an auditory interface. If users do not satisfy the retrieval results, users do over again the relevance feedback from the previous query or the previous results until they get music files they want. In the relevance feedback phase, we do range searching within in the representative melody index. The retrieval range of the relevance feedback is adjusted according to the degree of user's satisfaction to the previous retrieval results.

4.2 Content-Based Music Retrieval Using the Representative Melody Index

For content-based music retrievals in the developed system, we choose 'Content-based Retrieval', a submenu of 'Query' of Fig. 2. Then, we can see the content-based query interface as shown in Fig. 9.



Ranking by Time Warping distance

Title	Motif No	P_Value	L_Value	Sig1	Sig2	Sig3	Sig4	Sig5	Sig6	Sig7	Sig8	Rank_TW
가를길	1	1.250000	2.290909	1.00	1.00	1.00	3.00	6.00	3.00	1.00	2.00	0.00
소나무	4	1.277778	2.266667	1.67	2.33	1.00	4.33	1.67	2.33	1.33	1.33	7.68
이슬1	10	1.071429	2.250000	5.00	5.00	5.00	5.00	6.00	4.50	1.00	1.00	11.50

Fig. 9. Query and Result of Content-based Music Retrieval with a Long Query Melody

In this interface, users can draw the music scores of queries with the selection of expected number of results at 'Result Num'. 'Play' and 'Stop' buttons are to start and to stop listening the query melody, respectively. After drawing a query melody, by click 'Query' button users can get the retrieval results as shown at the lower window of Fig. 9. The retrieval results in the lower window of Fig. 9 come from the representative melody index. However, when we select a row in the results, we can see the full music score that is stored in the underlying music database as shown in Fig. 10. By clicking 'Play all' or 'Play selection' buttons, users can listen a selected part or full of music shown in music score window, respectively. After viewing and listening the retrieval results, users can advance the relevance feedback phase for a selected result at the lower window of Fig. 9 by clicking 'Feedback' button.

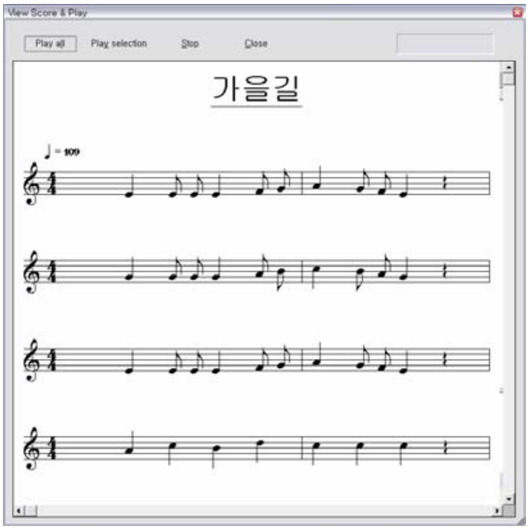
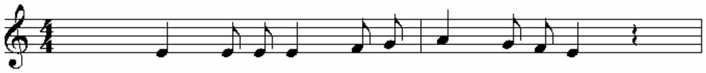
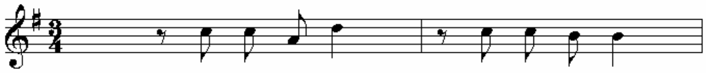
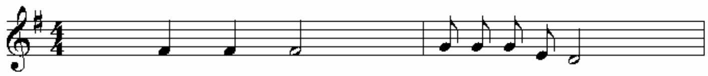


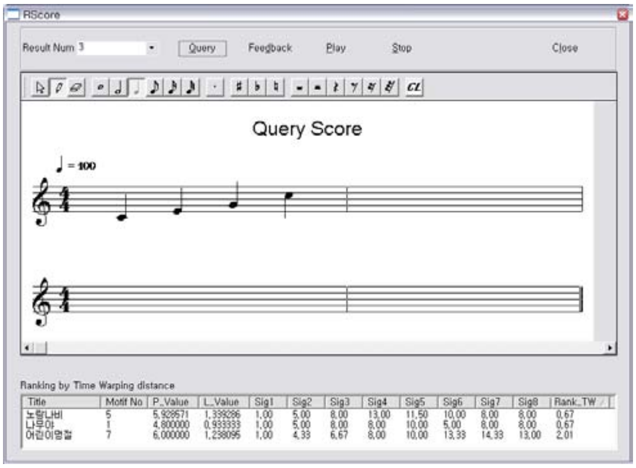
Fig. 10. Viewing and Listening Interface for Retrieval Results

To validate the appropriateness of the retrieval results from the representative melody index, we list up the retrieval results from the query melody of Fig. 9 at Table 1. From Table 1, we can easily recognize that the first result melody has exactly same music pattern to the prefix of the query melody given in Fig. 9, and also other retrieval results have also similar music pattern to the query melody.

Table 1. Retrieval Results for Long Query Melody of Fig. 9

Rank	Music Score
1	
2	
3	

When we submit a query of short length as shown in Fig. 11, we get three relevant melodies from the representative melody index. Also we can summarize the retrieval result as shown in Table 2. In similar way, we can validate the appropriateness of the retrieval results of the content-based music retrieval system using the representative melody index. From these validations, we know that the content-based music retrieval system developed in this work can return the appropriate results against query melodies even though they do not have same length to the melodies of the representative melody index, i.e., the lengths of queries are not one motif.

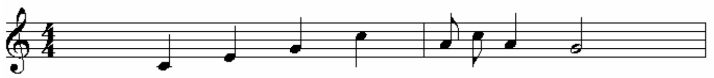
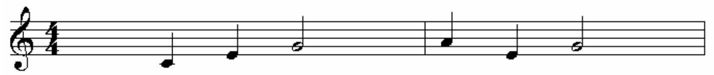
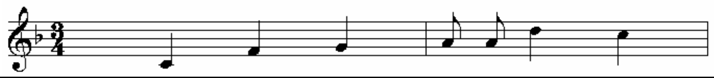


Ranking by Time Warping distance

Title	Motif No	P_Value	L_Value	Sig1	Sig2	Sig3	Sig4	Sig5	Sig6	Sig7	Sig8	Rank_TW
노랑나비	5	5.928571	1.339286	1.00	5.00	8.00	13.00	11.50	10.00	8.00	8.00	0.67
나무야	1	4.800000	0.933333	1.00	5.00	8.00	8.00	10.00	5.00	8.00	8.00	0.67
어린이명절	7	6.000000	1.238095	1.00	4.33	6.67	8.00	10.00	13.33	14.33	13.00	2.01

Fig. 11. Query and Result of Content-based Music Retrieval with a Short Query Melody

Table 2. Retrieval Results for Short Query Mellody of Fig. 11

Rank	Music Score
1	
2	
3	

4.3 Performance Evaluation

We did experiments with a music database of 265 Korean children songs. From the experimental database, we have representative melodies at total 660, whereas the total number of motifs in the database is 1,910. After construction of the representative melody index with 660 motifs for the experimental database, we do querying with the several types of query melodies, the first motif, the climax motif, and approximately repeated theme melody, respectively.

As the results, we can summarize that the supportable query types according to the index types. If we construct the representative melody index with only the first motifs of the underlying music database, we cannot retrieve effectively the relevant music by using other types of melodies such as the climax melodies and the approximately repeated theme melodies. If we use only the theme melody index, we cannot get the appropriate results against queries with the first melody and the climax melody. However, in the both cases of by using all representative melodies or by using whole melodies of the database, three kinds of queries can be supported effectively. However, we can achieve this effectiveness in the case of by using all representative melodies with smaller storage(about 65%) than in the case of whole melodies. Of course, if we use the whole melody index instead of the representative melody index, the system can support more affluence query types in which no representative melodies are used. In general, however, since users remember a music file by recalling its representative melodies, users hardly submit queries with no representative melodies.

Table 3. Supportable query types

Query melody / Index melody	First melody	Climax melody	Repeated theme melody
Only the first melodies	Supportable	Not supportable	Not supportable
Theme melodies	Not supportable	Not supportable	Supportable
Representative melodies	Supportable	Supportable	Supportable
Whole melodies	Supportable	Supportable	Supportable

5 Conclusions

In this paper, we discuss the design and implementation of a content-based music retrieval system in which the representative melody index is managed and used to improve the appropriateness and the performance of retrieval from users' melody queries. To design the prototype system, we first summarize the essential requirements for content-based music retrieval systems. Then, we introduce the overall architecture of the content-based music retrieval system developed in this work. We also discuss the construction of the representative melody index from the extracted melodies and the content-based retrieval procedure from users' melody queries. According to the experimental results, the system can save the index space up to 65% than the case of using the whole melody index while almost types of users queries can be supported.

In the content-based music retrieval system developed in this work, since the dimension of the metric space for the representative melody index is just 2, several melodies that have totally different music pattern may be placed within close distance. Even though we use the signatures of melodies to distinguish these melodies, we have slightly inappropriate results from melody queries. As the future work, therefore, we will design and implement a mapping function of music fragments into high-dimensional metric space for the representative melody index. After then, by using the system, we will develop a web-based e-commerce system for digitized music.

Acknowledgements

This work was supported by the Korea Science and Engineering Foundation(KOSEF) through the Northeast Asian e-Logistics Research Center at University of Incheon.

References

1. Niblack, W., et al.: QBIC Project: Querying Images by Content, Using Color, Texture, and Shape. Proceedings of the Conference on Storage and Retrieval for Image and Video Database. (1993).
2. Back, J. R.: The Virage Image Search Engine: An Open Framework for Image Management. Proceedings of the Conference on Storage and Retrieval for Image and Video Database. (1996).
3. Lu, G.: Indexing and Retrieval of Audio: A Survey. Journal of Multimedia Tools and Applications. 5 (2001) 269-290.
4. Kim, S., Kim, Y.: An Indexing and Retrieval Mechanism Using Representative Melodies for Music Databases. Proceedings of 2000 International Conference on Information Society in the 21st Century. (2000) 393-400.
5. Ghias, A., Logan, J., Charmberlin, D., Smith, B. C.: Query By Humming Musical Information Retrieval in an Audio Database. Proceedings of ACM Multimedia, (1995).
6. McNab, R., Smith, L, Witten, I.: Towards the Digital Music Library: Tune Retrieval from Acoustic Input. Proceedings of Digital Libraries, (1996).
7. Chou, T., Chen, A., Liu, C.: Music Databases: Indexing Techniques and Implementation. Proceedings of IEEE International Workshop on Multimedia Database Management Systems. (1996).
8. Hurson, D.: Themefinder. http://www.themefinder.com.
9. Meyer, L.: Explaining Music. Saegwang Publishing Co. (1990).
10. Jee, J., Oh, H.: Design and Implementation of Music Information Retrieval System Using Melodies. Journal of Korea Information Processing Society, Vol. 5(1). (1998).
11. Chen, A., Hsu J., Liu, C. C.: Efficient Repeating Pattern Finding in Music Databases. Proceedings of ACM Conference on Information and Knowledge Management. (1998).
12. Liu, C. C., Hsu, J. L., Chen, L. P.: Efficient Theme and Non-trivial Repeating Pattern Discovering in Music Databases. Proceedings of the 15th International Conference on Data Engineering. (1999).
13. Kang, Y., Ku, K., Kim, Y. S.: Extracting Theme Melodies by Using a Graphical Clustering Algorithm for Content-based Music Information Retrieval. Lecture Notes in Computer Science, Vol. 2151. Springer-Verlag. (2001).
14. Ku, K. Kim, Y. S.: Extraction of Representative Melodies Considering Musical Composition Forms for Content-based Music Retrievals. Proceedings of the IASTED International Conference on Databases and Applications. (2004).
15. Lee, B., Paek, G.: Everyone can compose songs. Jackeunwoori Publishing Co. Korea. (2000).
16. Lim, S., Ku, K., Kim, Y. S.: Similarity Computation between Music Motifs Using Cosine Measure. Proceedings of Korea Information Processing Society, Vol. 10(1). (2003).
17. Ciaccia, P., Patella, M., Zezula, P.: M-tree: An Efficient Access Method for Similarity Search in Metric Spaces. Proceedings of 23rd International Conference on VLDB. (1997).
18. Ha, J., Ku, K., Park, J., Kim, Y. S.: Construction of Theme Melody Index by Transforming Melody to Time-series Data for Content-based Music Information Retrieval. Journal of Korea Information Processing Society, Vol. 10. (2003).
19. Yi, B., Jagadish, H., Faloutsos, C.: Efficient Retrieval of Similar Time Sequences under Time Warping. Proceedings of IEEE International Conference on Data Engineering. (1998).

Methods for Combining Statistical Models of Music

Marcus Pearce, Darrell Conklin, and Geraint Wiggins

Centre for Computational Creativity, City University,
Northampton Square, London EC1V 0HB, UK
{m.t.pearce,conklin,geraint}@city.ac.uk

Abstract. The paper concerns the use of multiple viewpoint representation schemes for prediction with statistical models of monophonic music. We present an experimental comparison of the performance of two techniques for combining predictions within the multiple viewpoint framework. The results demonstrate that a new technique based on a weighted geometric mean outperforms existing techniques. This finding is discussed in terms of previous research in machine learning.

1 Introduction

Statistical models of symbolically represented music have been used in a number of theoretical and practical applications in the computer modelling and retrieval of music. Examples of such applications include computer-assisted composition [1, 2, 3], machine improvisation with human performers [4, 5], music information retrieval [6], stylistic analysis of music [7, 8, 9] and cognitive modelling of music perception [10, 11]. A significant challenge faced in much of this research arises from the need to simultaneously represent and process many different features or attributes of the musical surface. One approach to this problem is to represent music within a framework that allows a musical object to be observed from *multiple viewpoints* [12, 13]. Multiple viewpoint modelling strategies take advantage of such a representational framework by deriving individual expert models for any given representational viewpoint and then combining the results obtained from each model. Here we consider multiple viewpoint systems from the perspective of statistical modelling and prediction of monophonic music. In particular, we are concerned with the evaluation of different methods for combining the predictions of different models in a multiple viewpoint system. To this end, we compare the performance of a previously reported combination technique based on a weighted arithmetic mean [14] with a new technique based on a weighted geometric mean.

Multiple viewpoint systems are a specific instance of a more general class of strategies in machine learning collectively known as *ensemble learning methods*. As noted in [15], ensemble methods can improve the performance of machine learning algorithms for three fundamental reasons. The first is statistical: with small amounts of training data it is often hard to obtain reliable performance measures for a single model. By combining a number of well performing models, we can reduce the risk of inadvertently selecting models whose performance does not generalise well to new examples. The second reason is computational: for learning algorithms which employ local search, combining models which search locally from different starting points in the hypothesis space can yield better performance than any of the individual models. The final reason is representational: a combination of learning models may allow the system to reach parts of the hypothesis space that the individual models would be unable, or extremely unlikely, to reach. The development of multiple viewpoint systems was motivated largely by representational concerns arising specifically in the context of computer modelling of music [13]. Although ensemble methods have typically been applied in classification problems, as opposed to the prediction problems studied here, we shall draw on that body of work as required.

The paper is structured as follows. In §2, we review the theory of multiple viewpoints as a representational formalism, describe how we may develop statistical models within the multiple viewpoint framework and present the entropy based performance metrics that we shall use to assess the performance of our models. In §3, we introduce the techniques for combining viewpoint predictions and the experimental procedure that we use to evaluate them is described in §4. The results of our experiments are presented and discussed in §5. Finally, in §6, we conclude by suggesting some directions for future research.

2 Background

2.1 Representing Music with Multiple Viewpoints

In this section, we review the representation language of the multiple viewpoint framework as developed in [13, 14]. The specific motivation in the development of the framework was to extend the application of statistical modelling techniques to domains, such as music, where events have an internal structure and are richly representable in languages other than the basic event language. Here we consider the framework only insofar as it applies to monophonic music. See [16] for extensions to accommodate the representation of homophonic and polyphonic music.

The framework takes as its *musical surface* [17] sequences of musical events which roughly correspond to individual notes as notated in a score. Each event consists of a finite set of descriptive variables or *basic attributes* each of which may assume a value drawn from some finite domain or alphabet. Each attribute describes an abstract property of events and is associated with a type, τ, which specifies the properties of that attribute (see Table 1). Each type is associated with a syntactic domain, $[\tau]$, denoting the set of all syntactically valid elements of that type. Each type is also supplied with an informal semantics by means of an associated semantic domain, $[\![\tau]\!]$, which denotes the set of possible meanings for elements of type τ and a function, $[\![.]\!]_\tau : [\tau] \rightarrow [\![\tau]\!]$, which returns the semantic interpretation of any element of type τ. The Cartesian product of the domains of n basic types $\tau_1, \ldots, \tau_n$ is referred to as the *event space*, ξ:

Table 1. Sets and functions associated with typed attributes

Symbol	Interpretation	Example
τ	A typed attribute	`cpitch`
$[\tau]$	Syntactic domain of τ	$\{60, \ldots, 72\}$
$\langle\tau\rangle$	Type set of τ	$\{\texttt{cpitch}\}$
$[\![\tau]\!]$	Semantic domain of τ	$\{\mathrm{C}_4, \mathrm{C}\sharp_4, \ldots, \mathrm{B}_4, \mathrm{C}_5\}$
$[\![.]\!]_\tau : [\tau] \rightarrow [\![\tau]\!]$	Semantic interpretation of $[\tau]$	$[\![60]\!]_{\texttt{cpitch}} = \mathrm{C}_4$
$\Psi_\tau : \xi^* \rightharpoonup [\tau]$	see text	see text

$$\xi = [\tau_1] \times [\tau_2] \times \ldots \times [\tau_n]$$

An *event* $e \in \xi$ is an instantiation of the attributes $\tau_1, \ldots, \tau_n$ and consists of an n-tuple in the event space. The event space ξ, therefore, denotes the set of all representable events and its cardinality, $|\xi|$, will be infinite if one or more of the attribute domains $[\tau_1], \ldots, [\tau_n]$ is infinite. We shall use the notation $e_i^j \in \xi^*$ to denote a sequence of events $e_i, \ldots, e_j$ where $j \geq i \in \mathbb{Z}^+$ and ξ^* denotes the set of all sequences composed of members of ξ including the empty sequence ε.

A *viewpoint* modelling a type τ is a partial function, $\Psi_\tau : \xi^* \rightharpoonup [\tau]$, which maps sequences of events onto elements of type τ.[1] Each viewpoint is associated with a *type set* $\langle\tau\rangle \subseteq \{\tau_1, \ldots, \tau_n\}$, stating which basic types the viewpoint is derived from and is, therefore, capable of predicting [14]. A collection of viewpoints forms a *multiple viewpoint system.* We now describe the nature of several distinct *classes* of viewpoint which may be defined.

Basic Viewpoints For *basic types*, those associated with basic attribute domains, Ψ_τ is simply a projection function [14] and $\langle\tau\rangle$ is a singleton set containing just the basic type itself. An example of a basic type is one which represents the chromatic pitch of an event in terms of MIDI note numbers (`cpitch`; see Table 1).

Derived Viewpoints A type that does not feature in the event space but which is derived from one or more basic types is called a *derived type*. The function Ψ_τ acts as a *selector* function for events, returning the appropriate attribute value when supplied with an event sequence [14]. Since the function is partial the result may be undefined (denoted by $\perp$) for a given event sequence. Many of the derived types implemented in [14] are inspired by the construction of quotient GISs in [18]. The motivation for constructing such types is to capture and model the rich variety of relational and descriptive terms in a musical language [14]. A viewpoint modelling a derived type is called a *derived* viewpoint and the

[1] While viewpoints were defined in [13] to additionally comprise a statistical model of sequences in $[\tau]^*$, here we consider viewpoints to be a purely representational formalism and maintain a clear distinction between our representation language and our modelling strategies.

types from which it is derived, and which it is capable of predicting, are given by the type set for that viewpoint. An example of a derived viewpoint is one which represents melodic intervals in the chromatic pitch domain. Given the basic type `cpitch` shown in Table 1, the derived viewpoint `cpint` [14] is defined by the function:

$$\Psi_{\texttt{cpint}}(e_1^j) = \begin{cases} \perp & \text{if } j = 1, \\ \Psi_{\texttt{cpitch}}(e_j) - \Psi_{\texttt{cpitch}}(e_{j-1}) & \text{otherwise.} \end{cases}$$

Linked Viewpoints A system of viewpoints modelling primitive types will have limited representational and predictive power due to its inability to represent any interactions between those individual types [13]. *Linked viewpoints* are an attempt to address this problem and were motivated by the direct product GISs described in [18]. A *product type* $\tau = \tau_1 \otimes \ldots \otimes \tau_n$ between n constituent types $\tau_1, \ldots, \tau_n$ has the following properties:

$$[\tau] = [\tau_1] \times \ldots \times [\tau_n]$$

$$\langle\tau\rangle = \bigcup_{k=1}^{n} \langle\tau_k\rangle$$

$$[\![\tau]\!] = [\![\tau_1]\!] \textit{ and } \ldots \textit{ and } [\![\tau_n]\!]$$

$$\Psi_\tau(e_1^j) = \begin{cases} \perp & \text{if } \Psi_{\tau_i}(e_1^j) \text{ is undefined for any } i \in \{1, \ldots, n\} \\ \langle \Psi_{\tau_1}(e_1^j), \ldots, \Psi_{\tau_n}(e_1^j)\rangle & \text{otherwise.} \end{cases}$$

A linked viewpoint is one which models a product type. Linked viewpoints add to the representation language the ability to represent disjunctions of conjunctions of attribute values (as opposed to simple disjunctions of attribute values). To give an example, it was found in [13] that a viewpoint linking melodic pitch interval with inter-onset interval (`cpint` $\otimes$ `ioi`) proved useful in modelling the chorale melodies harmonised by J. S. Bach. This finding suggests that these two attributes types are correlated in that corpus.

Test Viewpoints A *test viewpoint* models a Boolean-valued attribute type and is used to define locations in a sequence of events [19]. An example is the `fib` viewpoint defined in [14] as follows:

$$\Psi_{\texttt{fib}}(e_1^j) = \begin{cases} \mathrm{T} & \text{if } \Psi_{\texttt{posinbar}}(e_1^j) = 1, \\ \mathrm{F} & \text{otherwise} \end{cases}$$

where `posinbar` is a derived type giving the relative position of an event in the bar (e.g., $[\![1]\!]_{\texttt{posinbar}}$ = the first event in the current bar).

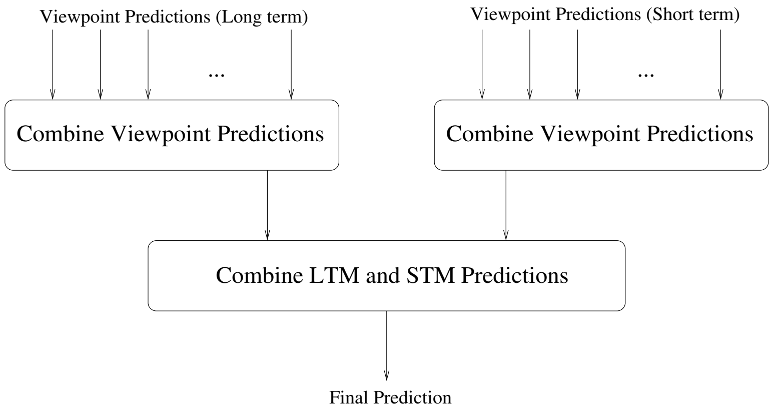

Viewpoint Predictions (Long term)
Viewpoint Predictions (Short term)
...
...
Combine Viewpoint Predictions
Combine Viewpoint Predictions
Combine LTM and STM Predictions
Final Prediction

Fig. 1. The architecture of a multiple viewpoint system

Threaded Viewpoints Types whose values are only defined at certain points in a piece of music (e.g., the first event in each bar) are called *threaded types* and viewpoints modelling these types are called *threaded viewpoints.* Threaded viewpoints model the value of a *base viewpoint* at temporal or metric locations where a specified test viewpoint returns true and are undefined otherwise [19]. The base viewpoint may be any primitive or linked viewpoint. Threaded viewpoints were developed to take advantage of structure emerging from metrical grouping and phrasing in music. The alphabet of a threaded viewpoint is the Cartesian product of the alphabets of the base viewpoint and a viewpoint, `ioi`, representing inter-onset intervals [19]. To take an example, consider the `thrbar` viewpoint defined in [13] constructed from the base viewpoint `cpint` and the test viewpoint `fib`. This viewpoint represents the melodic intervals between the first events in each consecutive bar and is undefined at all other locations in a melodic sequence. Its viewpoint elements consist of pairs of `cpint` and `ioi` elements corresponding to the pitch interval between the first events in two successive bars and the inter-onset interval between those events.

2.2 Modelling Music with Multiple Viewpoints

The Overall Architecture For our purposes, a statistical model associated with a viewpoint τ is a function m_τ which accepts a sequence of events in τ^* and which returns a distribution over $[\tau]$ reflecting the estimated conditional probabilities of the identity of the next viewpoint element in the sequence (see [13, 20] for further description of the nature of such models). Examples of such models include n-gram models which have been used for automatic classification of musical works [9], polyphonic score retrieval [6] and modelling of music perception [10, 11], and dictionary based statistical models which have been used for automatic music classification [7], computer improvisation with human performers [4] and computer-assisted composition [2].

A predictive system operating on a multiple viewpoint representation language consists of a number of models $m_{\tau_1}, \ldots, m_{\tau_n}$ corresponding to the collection of viewpoints $\tau_1, \ldots, \tau_n$ in the multiple viewpoint system. For each viewpoint, we actually employ two models: a *long-term model* (LTM) and a *short-term model* (STM). The LTM is trained on the entire training corpus while the STM is constructed online for each composition modelled and is discarded after the relevant composition has been processed. The motivation for using an STM is to take advantage of recently occurring sequences whose structure and statistics may be specific to the individual composition being predicted. The use of an STM has been found to improve the prediction performance of multiple viewpoint models of music [14, 20]. The predictions of both long- and short-term models must be combined to produce a final prediction (see §3). A number of general architectures can be envisaged to achieve this combination:

1. combine the STM and LTM predictions for each viewpoint individually and then combine the resulting viewpoint predictions;
2. combine the viewpoint predictions separately for the long- and short-term models and then combine the resulting LTM and STM predictions;
3. combine all long- and short-term viewpoint predictions in a single step.

We follow previous research [13] in choosing the second of these alternatives (see Figure 1). Two additional issues arise from the fact that our models accept sequences in $[\tau]^*$ rather than ξ^* and return distributions over $[\tau]$ rather than ξ: first, the corpus of event sequences in ξ^* must be preprocessed into sequences in τ^* which are used to train the models; and second, the resulting distribution over $[\tau]$ must be postprocessed into a distribution over ξ so it may be combined with distributions generated by other models. These issues are discussed in turn.

Preprocessing the Event Sequences We may convert sequences in ξ^* to sequences in $[\tau]^*$ using the function $\Phi_\tau : \xi^* \rightarrow [\tau]^*$ [13] such that:

$$\Phi_\tau(e_1^i) = \begin{cases} \varepsilon & \text{if } e_1^i = \varepsilon \\ \Phi_\tau(e_1^{i-1}) & \text{if } \Psi_\tau(e_1^i) = \perp \\ \Phi_\tau(e_1^{i-1})\Psi_\tau(e_i) & \text{otherwise} \end{cases}$$

Since $\Psi_\tau(e_1^i) = \perp \Rightarrow \Phi_\tau(e_1^i) = \Phi_\tau(e_1^{i-1})$, it is necessary to check that $\Psi_\tau(e_1^i)$ is defined to prevent the same sequence in $[\tau]^*$ being added to the model more than once [13].

Completion of a Multiple Viewpoint System A model m_τ returns a distribution over $[\tau]$ but, in order to combine the distributions generated by the models for different viewpoints, we need to convert them into distributions over the basic event space ξ. In the interests of efficiency, prediction is elicited in stages, one for each basic type of interest [14]. Only those viewpoints which contain in their type set the basic type, τ_b, currently under consideration are activated at each stage. The conversion is achieved by a function which maps elements of $[\tau]$ onto elements of $[\tau_b]$:

$$\Psi'_\tau : \xi^* \times [\tau] \to P([\tau_b])$$

where $P(S)$ denotes the power set of set S. The function Ψ'_τ is implemented by creating a set of events each of which corresponds to a distinct basic element in $[\tau_b]$. A set of sequences is created by appending each of these events to the sequence of previously processed events in the composition. By calling the function Ψ_τ on each of these sequences each element in $[\tau_b]$ is put into the mapping with the current element of $[\tau]$. The mapping is, in general, many-to-one since a derived sequence $\Phi_\tau(e_1^i)$ could represent many sequences of events other than e_1^i. As a result, the probability estimate returned by the model for the derived sequence must be divided equally among the basic event sequences onto which it maps.

A model m_τ must return a complete distribution over the basic attributes in $\langle\tau\rangle$. This does not present problems for basic viewpoints where the viewpoint domain is predefined to be the set of viewpoint elements occurring in the corpus.[2] However, for derived viewpoints, such as `cpint`, it may not be possible to derive a complete distribution over [`cpitch`] from the set of derived elements occurring in the corpus. To address this problem, the domain of each derived type τ is set prior to prediction of each event such that there is a one-to-one correspondence between $[\tau]$ and the domain of the basic type $\tau_b \in \langle\tau\rangle$ currently being predicted. We assume that the modelling technique has some facility for assigning probabilities to events that have never occurred before [13, 20]. If no viewpoints predict some basic attribute then the completion of that attribute must be predicted on the basis of information from other sources or on the basis of a uniform distribution over the attribute domain. In this research, m_{τ_b} was used to achieve the completion of attribute τ_b in such cases.

Once the distributions generated by each model in a multiple viewpoint system have been converted to complete distributions over the domain of a basic type, the distributions may be combined into final distributions for each basic type. The topic of this paper is how best to achieve this combination and two methods are discussed in detail in §3.

2.3 Performance Metrics

Given a probability mass function $p(a \in \mathcal{A}) = P(\mathcal{X} = a)$ of a random variable $\mathcal{X}$ distributed over a discrete alphabet $\mathcal{A}$, the entropy is calculated as:

$$H(p) = H(\mathcal{X}) = -\sum_{a\in\mathcal{A}} p(a)\log_2 p(a). \tag{1}$$

[2] The domain of a viewpoint modelling the onset time of events is potentially infinite and assumes a value derived from the onset time of the previous event and the set of inter-onset intervals that occur in the corpus [13].

Shannon's (1948) fundamental coding theorem states that entropy provides a lower bound on the average number of binary bits per symbol required to encode an outcome of the variable $\mathcal{X}$. The corresponding upper bound occurs in the case where each symbol in the alphabet has an equal probability of occurring, $\forall a \in \mathcal{A}, p(a) = \frac{1}{|\mathcal{A}|}$, as shown in Equation 2.

$$H_{max}(p) = H_{max}(\mathcal{A}) = \log_2 |\mathcal{A}| \tag{2}$$

Entropy has an alternative interpretation in terms of the degree of uncertainty that is involved in selecting a symbol from an alphabet: greater entropy implies greater uncertainty. In practise, we rarely know the true probability distribution of the stochastic process and use a model to approximate the probabilities in Equation 1. *Cross entropy* is a quantity which represents the divergence between the entropy calculated from these estimated probabilities and the source entropy. Given a model which assigns a probability of $p_m(a_1^j)$ to a sequence a_1^j of outcomes of $\mathcal{X}$, we can calculate the cross entropy $H(p_m, a_1^j)$ of model m with respect to event sequence a_1^j as shown in Equation 3.

$$H(p_m, a_1^j) = -\frac{1}{j}\sum_{i=1}^{j} \log_2 p_m(a_i|a_1^{i-1}) \tag{3}$$

While cross entropy provides a direct measure of performance in the field of data compression, it has a wider use in the evaluation of statistical models. Since it provides us with a measure of how uncertain a model is, on average, when predicting a given sequence of events, it can be used to compare the performance of different models on some corpus of data [21, 22].

3 Combining Viewpoint Prediction Probabilities

3.1 Introduction

In this section, we shall describe several techniques for combining the distributions generated by statistical models for different viewpoints. Let τ_b be the basic viewpoint currently under consideration and $[\tau_b] = \{t_1, t_2, \ldots, t_k\}$ its domain. Our multiple viewpoint system has n viewpoints $\tau_1, \ldots, \tau_n$ which are derived from τ_b and there exist corresponding sets of long-term models $LTM = \{ltm_1, ltm_2, \ldots, ltm_n\}$ and short-term models $STM = \{stm_1, stm_2, \ldots, stm_n\}$. We require a function that combines the distributions over τ_b generated by sets of models. As described in §2.2, this function is used in the first stage of combination to combine the distributions generated by the LTM and the STM separately and, in the second stage of prediction, to combine the two combined distributions resulting from the first stage. In what follows we describe functions for combining individual probabilities which may then be applied to sorted distributions over τ_b. For the purposes of illustration, we employ an anonymous set of models $M = m_1, m_2, \dots, m_n$.[3]

3.2 Arithmetic Combination

Perhaps the simplest method of combining distributions is to compute the arithmetic mean of the estimated probabilities for each symbol $t \in [\tau_b]$ such that:

$$p(t) = \frac{1}{n} \sum_{m \in M} p_m(t).$$

This combination technique may be improved by weighting the contributions made by each of the models such that:

$$p(t) = \frac{\sum_{m \in M} w_m p_m(t)}{\sum_{m \in M} w_m}.$$

A method for calculating the weights, w_m, is described in [14]. It is based on the entropies of the distributions generated by the individual models such that greater entropy (and hence uncertainty) is associated with a lower weight. The weight of model m is $w_m = H_{relative}(p_m)^{-b}$. The *relative entropy* $H_{relative}(p_m)$ of a model is given by:

$$H_{relative}(p_m) = \begin{cases} H(p_m)/H_{max}(p_m) & \text{if } H_{max}([\tau_b]) > 0 \\ 1 & \text{otherwise.} \end{cases}$$

where H and H_{max} are as defined in Equations 1 and 2 respectively. The bias $b \in \mathbb{Z}$ is a parameter giving an exponential bias towards models with lower relative entropy. Note that with $b = 0$, the weighted arithmetic scheme is equivalent to its non-weighted counterpart. This weighting mechanism is described in more detail in [14] where the weighted arithmetic mean was used for combining both viewpoint predictions and the predictions of the long- and short-term models while this method was used for combining viewpoint predictions only in [13].[4]

[3] We refer to combination schemes based on the arithmetic mean as arithmetic combination and those based on the geometric mean as geometric combination. Similar distinctions have been made in the literature between linear and logarithmic opinion pools [23], combining classifiers by averaging and multiplying [24] and mixtures and products of experts [25, 26].

[4] Other methods were also examined in [13] including a ranking-based combination method as well as a method based on the rule of combination used in the Dempster-Shafer theory of evidence.

Table 2. The basic and derived viewpoints used in this research

τ	Class	$[\![\cdot]\!]_\tau$	$[\tau]$	$\langle\tau\rangle$
`onset`	basic	onset time of event	{0,1,2,. . . }	{`onset`}
`cpitch`	basic	chromatic pitch (MIDI)	{60,. . . ,79,81}	{`cpitch`}
`ioi`	derived	inter-onset interval	{1,. . . ,20}	{`onset`}
`fib`	derived	(not) first event in bar	{T,F}	{`onset`}
`cpint`	derived	sequential melodic interval	$\mathbb{Z}$	{`cpitch`}
`cpintfref`	derived	vertical interval from referent	{0,. . . ,11}	{`cpitch`}

3.3 Geometric Combination

We present a novel method for combining the distributions generated by our statistical models which is based on a weighted geometric mean. A simple geometric mean of the estimated probabilities for each symbol $t \in [\tau_b]$ is calculated as:

$$p(t) = \frac{1}{R} \prod_{m \in M} p_m(t)^{\frac{1}{n}} .$$

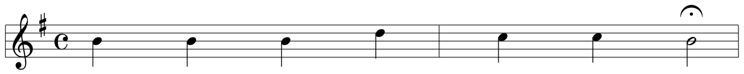
where R is a normalisation constant. As in the case of the arithmetic mean, this technique may be improved by weighting the contributions made by each of the models such that:

$$p(t) = \frac{1}{R} \prod_{m \in M} p_m(t)^{w_m}$$

where R is a normalisation constant and the weights w_m are normalised such that they sum to one. We may use the same weighting technique as for arithmetic combination (see §3.2) and, once again, with $b = 0$, the weighted geometric scheme is equivalent to its non-weighted counterpart.

4 Experimental Procedure

The corpus of music used is a subset of the chorale melodies harmonised by J. S. Bach. A set of 185 chorales (BWV 253 to BWV 438) has been encoded by Steven Rasmussen and is freely available in the `**kern` format [27] from the *Centre for Computer Assisted Research in the Humanities* (CCARH) at Stanford University, California (see http://www.ccarh.org). We have used cross entropy, as defined in Equation 3, computed by 10-fold cross-validation [28, 29] over the corpus as a performance metric for our models. The statistical model used was a smoothed, variable-order n-gram model described in more detail in [20]. Since the goal of this research was to examine methods for combining



`onset`	0	24	48	72	96	120	144
`cpitch`	71	71	71	74	72	72	71
`ioi`	$\perp$	24	24	24	24	24	24
`fib`	T	F	F	F	T	F	F
`cpint`	$\perp$	0	0	3	-2	0	-1
`cpintfref`	4	4	4	7	5	5	4
`cpint`$\otimes$`ioi`	$\perp$	(0 24)	(0 24)	(3 24)	(-2 24)	(0 24)	(-1 24)

Fig. 2. The first phrase of the chorale melody *Meinen Jesum laß' ich nicht, Jesus* (BWV 379) represented as viewpoint sequences in terms of the basic, derived and linked viewpoints used in the experiments

viewpoint predictions, we have used a constant set of viewpoints corresponding to the best performing of the multiple viewpoint systems described in [13]. This system consists of the following viewpoints:

$$\texttt{cpintfref} \otimes \texttt{cpint},$$
$$\texttt{cpint} \otimes \texttt{ioi},$$
$$\texttt{cpitch},$$
$$\texttt{cpintfref} \otimes \texttt{fib}.$$

It is capable of modelling the basic type `cpitch` alone. See Table 2 for details of each of the viewpoints in this system and Figure 5 for an exemplary use of these viewpoints in representing an excerpt from a chorale melody in terms of viewpoint sequences. We have examined the weighted arithmetic and geometric combination schemes described in §3 in both stages of combination with the bias settings drawn from the set {0,1,2,3,4,5,6,7,8,16,32}.[5]

5 Results and Discussion

The results of the experiment are shown in Table 3 which is divided into four sections corresponding to the four combinations of the two combination methods. Figures in bold type represent the lowest entropies in each of the four sections of the table. The results are also plotted graphically in Figure 5. The first point to note is that the multiple viewpoint system is capable of predicting the dataset

[5] The Dempster-Shafer and rank-based combination schemes described in [13] were found to perform less well than these two methods (when optimally weighted) and are not included in the results.

with much lower entropies (e.g., 2.045 bits/symbol) than those reported in [20] for a system modelling chromatic pitch alone (e.g., 2.342 bits/symbol) on the same corpus. This replicates the findings of [13] and lends support to the assertion that the multiple viewpoint framework can increase the predictive power of statistical models of music. It is also clear that the use of an entropy based weighting scheme improves performance and that performance can be further improved by tuning the bias parameter which gives exponential bias towards models with lower relative entropies [14].

Regarding the combination methods, the results demonstrate that the weighted geometric combination introduced in this paper tends to outperform arithmetic combination and that this effect is much more marked in the case of viewpoint combination than it is for LTM-STM combination. Some theoretical justification for this result can be found in the literature on combining classifier systems. Hinton [25, 26] argues that combining distributions through multiplication has the attractive property of making distributions "sharper" than the component distributions. For a given element of the distributions it suffices for just one model to correctly assign that element a low estimated probability. If this is the case, the combined distribution will assign that element a low probability regardless of whether other models (incorrectly) assign that element a high estimated probability. Arithmetic combination, on the other hand, will tend to produce combined distributions that are less sharp than the component distributions and is prone to erroneously assigning relatively high estimated probabilities to irrelevant elements. However, since the combined distribution cannot be sharper than any of the component distributions arithmetic combination has the desirable effect of suppressing estimation errors [24].

In [24] the performance of arithmetic and geometric combination schemes is examined in the context of multiple classifier systems. In accordance with theoretical predictions, an arithmetic scheme performs better when the classifiers operate on identical data representations and a geometric scheme performs better when the classifiers employ independent data representations. Analogously, we hypothesise that when combining viewpoint predictions (derived from distinct data representations), a geometric scheme performs better since it trusts specialised viewpoints to correctly assign low probability estimates to a given element. Consider movement to a non scale degree as an example: a model associated with `cpitch` might return a high probability estimate for such a transition whereas a model associated with `cpintfref` is likely to return a low estimated probability. In cases such as this, it is preferable to trust the model operating over the more specialised data representation (i.e., `cpintfref`).

When combining LTM-STM predictions (where each distribution is already the result of combining the viewpoint predictions), on the other hand, a premium is placed on minimising estimation errors. For example, for n-grams which are common in the current composition but rare in the corpus as a whole, the LTM will return low estimates and the STM high estimates. In cases such as this, it is preferable to suppress the estimation errors yielded by the LTM. The finding that geometric combination still outperforms arithmetic combination in LTM-

Table 3. Cross entropies (bits/symbol) of the data given the model using weighted arithmetic and geometric schemes with a range of bias settings for combining the LTM-STM and viewpoint predictions

		Viewpoint Combination																					
		Arithmetic											Geometric										
		0	1	2	3	4	5	6	7	8	16	32	0	1	2	3	4	5	6	7	8	16	32
A	0	2.493	2.437	2.393	2.363	2.342	2.327	2.316	2.309	2.304	2.290	2.291	2.357	2.321	2.299	2.286	2.278	2.274	2.271	2.270	2.269	2.274	2.285
r	1	2.434	2.368	2.317	2.281	2.257	2.241	2.230	2.222	2.217	2.207	2.212	2.256	2.216	2.192	2.180	2.175	2.173	2.173	2.174	2.175	2.188	2.203
i	2	2.386	2.317	2.264	2.229	2.207	2.193	2.184	2.178	2.175	2.171	2.178	2.189	2.150	2.130	2.123	2.122	2.124	2.126	2.130	2.133	2.152	2.169
t	3	2.350	2.279	2.228	2.196	2.177	2.166	2.160	2.156	2.155	2.159	2.168	2.146	2.111	2.096	2.094	2.097	2.102	2.107	2.112	2.117	2.142	2.160
h	4	2.323	2.253	2.204	2.175	2.159	2.150	2.147	2.145	2.146	2.157	2.169	2.119	2.088	2.077	2.078	2.085	2.092	2.100	2.107	2.113	2.142	2.161
m	5	2.303	2.234	2.188	2.161	2.147	2.141	2.139	2.140	2.141	2.158	2.173	2.102	2.074	2.066	2.070	2.079	2.089	2.098	2.106	2.113	2.146	2.167
e	6	2.288	2.221	2.176	2.152	2.140	2.136	2.135	2.137	2.140	2.161	2.179	2.091	2.066	2.060	2.066	2.077	2.088	2.098	2.108	2.116	2.151	2.174
t	7	2.276	2.211	2.168	2.146	2.136	2.133	2.134	2.136	2.140	2.165	2.184	2.085	2.061	2.057	2.064	2.076	2.088	2.099	2.110	2.118	2.156	2.180
i	8	2.268	2.204	2.163	2.142	2.133	**2.131**	2.133	2.136	2.140	2.168	2.189	2.080	2.057	2.055	2.064	2.076	2.089	2.101	2.112	2.121	2.161	2.186
c	16	2.243	2.186	2.152	2.136	2.132	2.133	2.138	2.143	2.149	2.184	2.212	2.073	**2.053**	2.054	2.066	2.081	2.097	2.111	2.123	2.134	2.182	2.212
	32	2.239	2.185	2.154	2.140	2.138	2.140	2.145	2.151	2.157	2.195	2.226	2.074	2.055	2.058	2.070	2.086	2.103	2.118	2.132	2.143	2.194	2.226
	0	2.496	2.437	2.386	2.346	2.316	2.294	2.278	2.266	2.257	2.237	2.240	2.311	2.267	2.238	2.222	2.213	2.208	2.207	2.206	2.207	2.219	2.234
G	1	2.425	2.354	2.295	2.252	2.222	2.202	2.188	2.178	2.172	2.160	2.165	2.200	2.155	2.129	2.118	2.114	2.114	2.116	2.119	2.122	2.141	2.157
e	2	2.372	2.298	2.240	2.201	2.176	2.161	2.151	2.145	2.142	2.142	2.150	2.138	2.098	2.081	2.077	2.079	2.084	2.090	2.096	2.101	2.126	2.143
o	3	2.334	2.260	2.206	2.172	2.152	2.141	2.135	2.133	2.132	2.141	2.154	2.104	2.070	2.059	2.060	2.067	2.076	2.084	2.092	2.099	2.13	2.149
m	4	2.307	2.235	2.185	2.155	2.139	2.131	2.128	2.128	2.129	2.146	2.163	2.086	2.057	2.050	2.054	2.064	2.075	2.085	2.095	2.103	2.138	2.159
e	5	2.288	2.218	2.171	2.145	2.132	2.127	2.126	2.127	2.130	2.152	2.171	2.077	2.051	2.046	2.053	2.065	2.077	2.089	2.099	2.108	2.146	2.169
t	6	2.275	2.207	2.163	2.139	2.129	**2.125**	2.126	2.128	2.132	2.158	2.179	2.072	2.048	**2.045**	2.054	2.067	2.080	2.092	2.103	2.113	2.154	2.178
r	7	2.265	2.200	2.158	2.136	2.127	**2.125**	2.127	2.130	2.134	2.163	2.186	2.069	2.047	**2.045**	2.055	2.069	2.083	2.096	2.107	2.117	2.160	2.185
i	8	2.258	2.194	2.155	2.134	2.127	**2.125**	2.128	2.132	2.136	2.167	2.192	2.068	2.047	2.046	2.057	2.071	2.085	2.099	2.111	2.121	2.165	2.191
c	16	2.240	2.184	2.151	2.136	2.132	2.133	2.138	2.144	2.150	2.186	2.216	2.070	2.051	2.053	2.065	2.081	2.098	2.112	2.125	2.136	2.186	2.217
	32	2.239	2.185	2.154	2.141	2.138	2.141	2.146	2.151	2.158	2.198	2.229	2.073	2.055	2.057	2.070	2.087	2.104	2.12	2.134	2.145	2.197	2.230

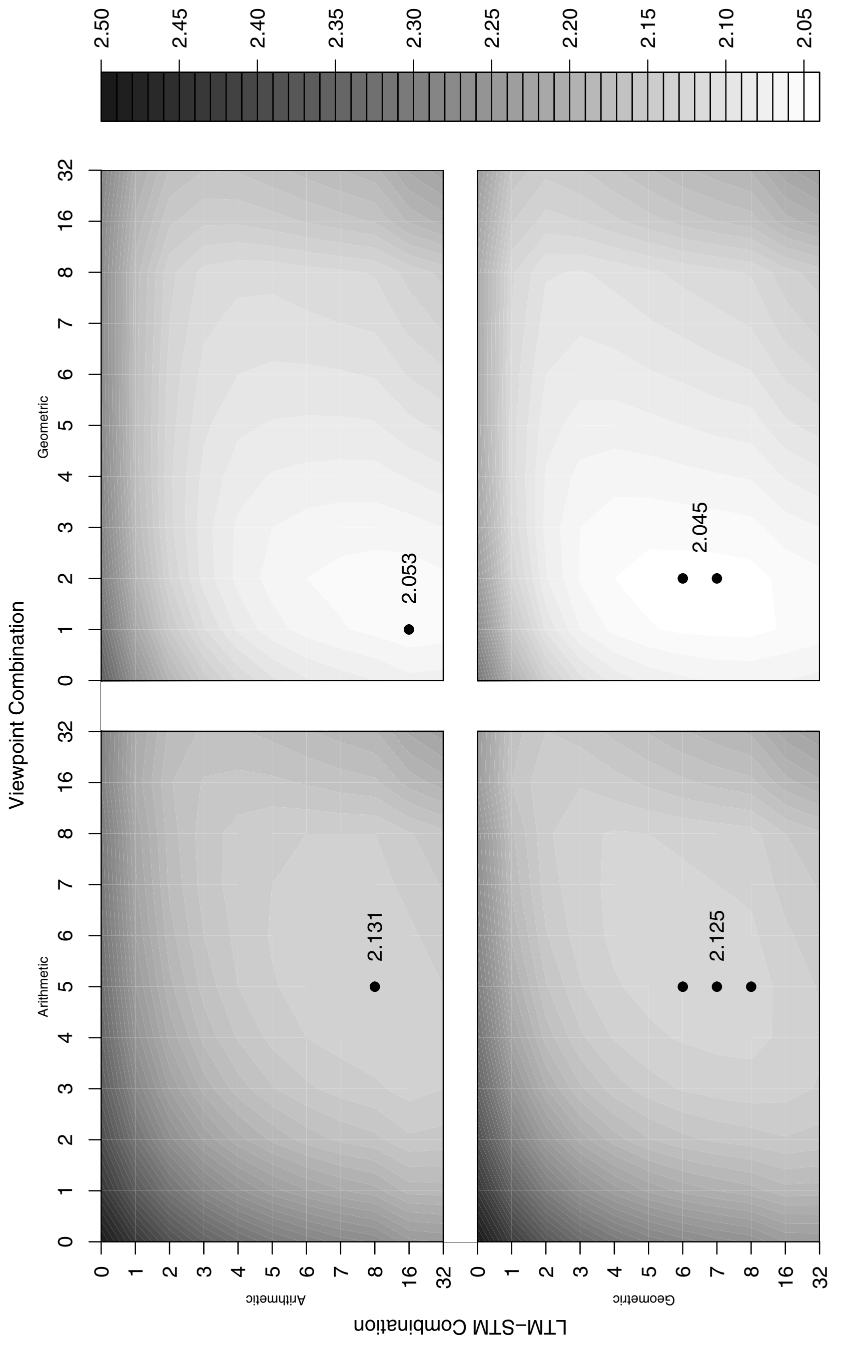


Fig. 3. Cross entropies (bits/symbol) of the data given the model using weighted arithmetic and geometric schemes with a range of bias settings for combining the LTM-STM and viewpoint predictions

STM combination may be a result of the fact that n-grams are added online to the LTM as prediction progresses much as they are for the STM [20]. Finally, it is possible that the difference in relative performance of the geometric and arithmetic schemes for LTM-STM and viewpoint combination is a result of the order in which these combinations are performed (see Figure 1). However, we hypothesise that this is not the case and the observed pattern of results arises from the difference between combining distributions derived from distinct data representations as opposed to combining two distributions already combined from the same sets of representations. Further research is required to examine these hypotheses in more depth.

Another aspect of the results that warrants discussion is the effect on performance of the bias parameter which gives an exponential bias towards distributions with lower relative entropy. Overall performance seems to be optimised when the bias for LTM-STM combination is relatively high (between 6 and 16) and the bias for viewpoint combination is relatively low (between 1 and 5). We suggest that this is due to the fact that at the beginning of a composition, the STM will generate relatively high entropy distributions due to the lack of context. In this case, it will be advantageous for the system to strongly bias the combination towards the LTM predictions. This is not an issue when combining viewpoint predictions and more moderate bias values tend to be optimal. Other research has also found that high bias values for the combination of the LTM-STM predictions tend to improve performance leading to the suggestion that the weight assigned to the STM could be progressively increased from an initially low value at the beginning of a composition as more events are processed [14].

The results shown in Table 2 also reveal an inverse relationship between the optimal bias settings for LTM-STM combination and those for viewpoint combination. With high bias values for LTM-STM combination, low bias values for viewpoint combination tend to be optimal and vice versa. High bias settings will make the system bolder in its estimation by strongly favouring sharper distributions while low bias settings will lead it to more conservative predictions. On these grounds, with all other things being equal, we would expect moderate bias values to yield optimal performance. If an extreme bias setting is preferred in one stage of combination for some other reason (e.g., the case of LTM-STM combination just discussed), the negative effects may, it seems, be counteracted to some extent by using settings at the opposing extreme in the other stage. Although these arguments are general, we would expect the optimal bias settings themselves to vary with different data, viewpoints and predictive systems.

6 Conclusions

We have presented an experimental comparison of the performance of two techniques for combining distributions within the multiple viewpoint framework for representing and modelling music. Specifically, a novel combination technique based on a weighted geometric mean was compared to an existing technique based on a weighted arithmetic mean. We have used an entropy based technique to compute the weights which accepts a parameter which fine-tunes the exponential bias given to distributions with lower relative entropy. A range of parameterisations of the two techniques have been evaluated using cross entropy computed by 10-fold cross-validation over a dataset of chorale melodies harmonised by J. S. Bach. The results demonstrate that the weighted geometric combination introduced in this research tends to outperform arithmetic combination especially for the combination of viewpoint models. Drawing on related findings in previous research in machine learning on combining multiple classifiers, it was hypothesised that this asymmetry arises from the difference between combining distributions derived from distinct data representations as opposed to combining distributions derived from the same data representations.

We would like to conclude the paper by suggesting some directions we feel would be fruitful for future research. Perhaps the most important limitation of this research is that results have been obtained for a single dataset (representing a single genre of melodic music) using a single set of viewpoints. However, this research does make specific hypotheses to be refuted or corroborated by further experiments which go beyond these restrictions. Our confidence in the generality of these results obtained would be increased if they could be replicated using different corpora of music, different viewpoint systems and other forms of music. It would also be useful to conduct a thorough examination of the effect of the overall architecture of the system on performance. How is performance affected, for example, if we first combine the LTM-STM predictions for each viewpoint and then combine the resulting distributions? It seems unlikely that a single combination of all distributions will improve performance but this conjecture can only be tested by empirical experimentation. Finally, it remains to be seen whether other combination schemes developed in the field of machine learning [30, 31, 32] can be profitably applied to modelling music with multiple viewpoint systems.

This research has examined a number of techniques for improving the prediction performance of statistical models of music. These techniques have been evaluated in an application neutral manner using cross entropy as an index of model uncertainty. In statistical language modelling, it has been demonstrated that cross entropy provides a good predictor of model performance in specific practical contexts:

> "For a number of natural language processing tasks, such as speech recognition, machine translation, handwriting recognition, stenotype transcription and spelling correction, language models for which the cross entropy is lower lead directly to better performance."
>
> [21, p. 39].

While corresponding results are not currently available in the literature on computational music research, we believe the techniques presented in this paper can be applied profitably to practical problems in the modelling and retrieval of music.

References

[1] Ames, C.: The Markov process as a compositional model: a survey and tutorial. Leonardo **22** (1989) 175–187

[2] Assayag, G., Dubnov, S., Delerue, O.: Guessing the composer's mind: applying universal prediction to musical style. In: Proceedings of the 1999 International Computer Music Conference, San Francisco: ICMA (1999) 496–499

[3] Hall, M., Smith, L.: A computer model of blues music and its evaluation. Journal of the Acoustical Society of America **100** (1996) 1163–1167

[4] Lartillot, O., Dubnov, S., Assayag, G., Bejerano, G.: Automatic modelling of musical style. In: Proceedings of the 2001 International Computer Music Conference, San Francisco: ICMA (2001) 447–454

[5] Rowe, R.J.: Machine composing and listening with Cypher. Computer Music Journal **16** (1992) 43–63

[6] Pickens, J., Bello, J.P., Monti, G., Sandler, M.B., Crawford, T., Dovey, M., Byrd, D.: Polyphonic score retrieval using polyphonic audio queries: A harmonic modeling approach. Journal of New Music Research **32** (2003) 223–236

[7] Dubnov, S., Assayag, G., El-Yaniv, R.: Universal classification applied to musical sequences. In: Proceedings of the 1998 International Computer Music Conference, San Francisco: ICMA (1998) 332–340

[8] Ponsford, D., Wiggins, G.A., Mellish, C.: Statistical learning of harmonic movement. Journal of New Music Research **28** (1999) 150–177

[9] Westhead, M.D., Smaill, A.: Automatic characterisation of musical style. In Smith, M., Smaill, A., Wiggins, G., eds.: Music Education: an Artificial Intelligence Approach, Berlin, Springer (1993) 157–170

[10] Ferrand, M., Nelson, P., Wiggins, G.: A probabilistic model for melody segmentation. In: Electronic Proceedings of the 2nd International Conference on Music and Artificial Intelligence (ICMAI'2002), University of Edinburgh, Scotland (2002)

[11] Eerola, T.: The Dynamics of Musical Expectancy: Cross-cultural and Statistical Approaches to Melodic Expectations. PhD thesis, Faculty of Humanities, University of Jyväskylä, Finland (2004) Jyväskylä Studies in Humanities, 9.

[12] Ebcioğlu, K.: An expert system for harmonising four–part chorales. Computer Music Journal **12** (1988) 43–51

[13] Conklin, D., Witten, I.H.: Multiple viewpoint systems for music prediction. Journal of New Music Research **24** (1995) 51–73

[14] Conklin, D.: Prediction and entropy of music. Master's thesis, Department of Computer Science, University of Calgary (1990) Available as Technical Report 1990–390–14.

[15] Dietterich, T.G.: Ensemble methods in machine learning. In: First International Workshop on Multiple Classifier Systems. Lecture Notes in Computer Science, New York: Springer Verlag (2000) 1–15

[16] Conklin, D.: Representation and discovery of vertical patterns in music. In Anagnostopoulou, C., Ferrand, M., Smaill, A., eds.: Proceedings of the Second International Conference of Music and Artificial Intelligence. (2002) 32–42

[17] Jackendoff, R.: Consciousness and the Computational Mind. MIT Press, Cambridge, MA (1987)

[18] Lewin, D.: Generalised Musical Intervals and Transformations. Yale University Press, New Haven/London (1987)

[19] Conklin, D., Anagnostopoulou, C.: Representation and discovery of multiple viewpoint patterns. In: Proceedings of the 2001 International Computer Music Conference, San Francisco: ICMA (2001)

[20] Pearce, M.T., Wiggins, G.A.: Improved methods for statistical modelling of monophonic music. To appear in *Journal of New Music Research* (2004)
[21] Brown, P.F., Della Pietra, S.A., Della Pietra, V.J., Lai, J.C., Mercer, R.L.: An estimate of an upper bound on the entropy of English. Computational Linguistics **18** (1992) 32–40
[22] Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge, MA (1999)
[23] Genest, C., Zidek, J.V.: Combining probability distributions: a critique and an annotated bibliography. Statistical Science **1** (1986) 114–148
[24] Tax, D.M.J., van Breukelen, M., Duin, R.P.W., Kittler, J.: Combining multiple classifiers by averaging or by multiplying? Pattern Recognition **33** (2000) 1475–1485
[25] Hinton, G.: Products of experts. In: Proceedings of the Ninth International Conference on Artificial Neural Networks. Volume 1. (1999) 1–6
[26] Hinton, G.: Training products of experts by minimizing contrastive divergence. Technical Report GCNU TR 2000-004, Gatsby Computational Neuroscience Unit, UCL (2000)
[27] Huron, D.: *Humdrum* and *Kern*: selective feature encoding. In Selfridge-Field, E., ed.: Beyond MIDI: The Handbook of Musical Codes. MIT Press, Cambridge, MA (1997) 375–401
[28] Mitchell, T.M.: Machine Learning. McGraw Hill, New York (1997)
[29] Dietterich, T.G.: Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation **10** (1998) 1895–1924
[30] Xu, L., Krzyzak, A., Suen, C.Y.: Methods of combining multiple classifiers and thier applications to handwriting recognition. IEEE Transactions on Systems, Man and Cybernetics **22** (1992) 418–435
[31] Chen, K., Wang, L., Chi, H.: Methods of combining multiple classifiers with different features and their applications to text-independent speaker identification. International Journal of Pattern Recognition and Artificial Intelligence **11** (1997) 417–415
[32] Kittler, J., Hatef, M., Duin, R.P.W., Matas, J.: On combining classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 226–239

Constraint-Based Melody Representation

Ningyan Zhong[1] and Yi Zheng[2]

[1] University of Waterloo, Ontario, Canada
nzhong@softbase.uwaterloo.ca
[2] University of Toronto, Ontario, Canada
yi.zheng@utoronto.ca

Abstract. A more natural way of searching a song is by humming or singing, especially when people can not remember the title, singer or lyric, but can remember a piece of melody. This paper is about how to represent melody so that such query is possible and efficient. We propose a constraint-based method. Considering both local and global constraints of pitch and duration, we can derive a melody's canonical form by using linear algorithm *Reduce* and *Weight*. This representation is at least as precise as interval-based method, but avoid much of redundant searching since it is more specialized. The significance of our method is that the tonal context, a key factor of music, is taken into account. This feature makes it not only a good candidate for building an efficient music database system, but also a foundation for further investigation of music, such as style analysis, pattern extraction, and finding other constraints.

1 Introduction to Constraint Database

Constraint Database (CDB)[18] is an evolution of Constraint Logic Programming (CLP) [10] and Relational Database[9], where we can encode the data set in some type of constraint formula. The data set, finite or infinite, is stored compactly which allows much more efficient operations. The key to CDB is that constraint itself can be viewed as the ground tuple and generalized to allow complex constraint over many attributes.

Kanellakis[16], defines CDB as follows:

– *A generalized k-tuple*: a finite conjunction of constraints $\varphi_1 \wedge \cdots \wedge \varphi_n$, where each φ_i $(i = 1, \cdots, n)$ is a constraint over variables $x_1, \cdots, x_k$.
– *A generalized relation of arity k*: a finite set $r = \{\psi_1, \cdots, \psi_m\}$ of *a generalized k-tuple* over the same variables.
– *A generalized database*: a finite set of *generalized relations*.

The formula corresponding to a *generalized relation* r is the disjunction $\psi_1 \vee \cdots \vee \psi_m$. We use ϕ_r to denote the quantifier-free formula corresponding to relation r. *Quantifier Elimination* (QE) is the removal of all quantifiers, $\exists$ and $\forall$, from a quantified formula. A First-Order theory allows QE if, for each quantified formula, there exists an equivalent quantifier-free formula.

From the above we can see that the tuple is defined as a conjunction of constraints. Under such a framework it is not easy to implement the algebra, or the database operations, particularly, *projection*. Therefore we define a new framework of CDB, basing on the canonical representation of constraints:

- *An atomic constraint* (component) c is the constraint formula over a set of variables $x_1, \cdots, x_k$ (in some cases, the order of variables are also needed to be considered), indicating the condition that these variables need to satisfy. We use $D(c)$ to denote the solutions of c. c normally has a fixed structure over fixed number of variables.
- *Canonical Form cf* is a finite set of *atomic constraint*s $\{c_1, \cdots, c_n\}$, where $D(c_i) \cap D(c_j) = \emptyset, 1 \leq i, j \leq n, i \neq j$. An important property of canonical form is that it represents the data set $D(cf) = D(c_1) \cup \cdots \cup D(c_n)$ uniquely: $cf_1 = cf_2 \Longleftrightarrow D(cf_1) = D(cf_2)$, $cf_1 \neq cf_2 \Longleftrightarrow D(cf_1) \neq D(cf_2)$.
- *A tuple* t is an atomic constraint, and a relation can be written as a disjunction $c_1 \vee \cdots \vee c_n$.

Canonical form is a succinct and precise representation of constraints. Finding canonical form of constraints usually involves techniques of constraint solving from CLP and symbolic computation.

An important property of canonical form representation is its *uniqueness*—there is only one representation for a fixed data set and we are able to distinguish two different sets. For example, the constraints $3x - 3y = 3$ and $2x - 2y = 2$ represent the same data set although they look different. However, if we define the canonical form of such constraints to be the form that y's coefficient is 1, these equivalent constraints will be reduced to only one formula $x - y = 1$. Due to uniqueness, the *equivalence* operation can be done by just checking whether two constraints have the same canonical representation. Based on canonical form of constraints, we can extend relational databases with certain type of constraints, and define constraint algebras.

Music is composed of a sequence of sounds. Do these sounds satisfy any kind of constraint? How to describe the constraint? Sect. 2 will give an introduction to some musical knowledge. Then Sect. 3 focuses on proving the canonical form for melody constraints by using *Reduce* and *Weight* algorithms. The experimental results are given in Sect. 4. Sect. 5 is the proposal for applying this representation in building music databases, and discussion of some further questions. Final section is the conclusion of the paper.

2 Basic Music Concepts

The music is usually taken as a piece of harmonic sound, and provides us with great entertainment. The following are several music concepts defined on technical level(refer to [8,17]):

Pitch. Pitch is related to the fundamental frequency of a sound. The higher the frequency, the higher the sound is.

Note. People use symbols like C D E F, etc., to represent pitch classes. For example, note A4 is defined to have a frequency of 440 Hz.

Octave. If one note has double the frequency of another, then it is said to be one *octave* above the lower note, and we usually use the same note to represent.
Semitone. An *octave* is divided into 12 parts evenly, which is called *semitone*s. *Wholetone* is the double of a semitone.
Interval. *Interval* is the number of semitones between two notes, e.g., the interval between C and D is 2.
Scale. Though an octave contains 12 notes, a *scale* only uses a subset of them. For example, the C major scale consists of the notes C D E F G A B C in ascending order of pitch's frequency, where the intervals between E and F, and between B and C are semitones, the others are wholetones. In other words, a scale contains semitone intervals at two fixed positions in one octave, while the others are wholetone intervals.
Sharp (#) and Flat(b) Symbols. Given a note, e.g., C, a note which is one semitone higher or lower than C can be written as $C^{\#}$ or C^{b} respectively.
Duration. The time that a note sounds. Usually it is measured in second.
Melody. Melody is the principal part in a harmonized piece of music, which helps people distinguish different music. A melody is mainly determined by the following two sequences of constants : the interval and the ratio of durations between two adjacent pitches.
Contour. A rough description of a melody. Notes can be reduced into a string format, such as USDSUUD, where U, D, and S represent whether or not a note is Up, Down or Same compared with the preceding note.
Music Notation Systems. Music notation, the way of writing down music, has developed over many years.

The ancient Greeks and Romans used letters of their alphabets to symbolize notes, from which came our use of the letters A to G to represent notes which is still common in many countries.

France, Italy and other associated countries tend to use the *tonic sol fa* names (based on C as Doh) as names of notes, rather than alphabetical letters. Most people know of it from the song "Doh re me", from the 1959 film "The Sound of Music" by Rogers and Hammerstein. However, the origins of the *tonic sol fa* are the eleventh century. A Benedictine monk, Guido of Arezzo, took the first notes of each line of a Latin hymn, written around A.D. 770, make them to be the first six notes of a major scale, and used the syllables of the Latin words, "doh", "re", "mi", "fa", "sol" and "la", that were sung on those notes. The seventh note "si" was added later because it was probably not a part of the normal scale at that time.

Modern notation (five-line *stave notation*) is much more precise than these older notations, developed initially in the fourteenth century and spread to the rest of the world. Nowadays *stave notation* becomes the dominant music language.

In China, the most popular music notation in elementary school education is *number notation* developed in the twentieth century, based on *tonic sol fa* system. That gives each note a number 1 2 3 4 5 6 7, where they are sung as "doh", "re", "mi", "fa", "sol", "la" and "si". A melody is represented by a sequence of numbers, together with some other symbols like duration of a note. In real singing, we need to adjust 1 to proper pitch, such as 1 =C. In this way we can change the tonal context from one to another easily. There is an interesting fact that, after some training, i.e., being more sensitive to music, people are able to translate a song into numbered notes just by listening several times, without any help from instruments. The number sequences of a melody translated by different people are almost same, and it doesn't matter which scale the melody is played. It seems that the relative information among pitches of a melody ensures a unique number sequence in general. Our idea is to translate pitches into numbers automatically and prove the correctness and uniqueness of the constraint-based representation. An example of stave and number notation systems for a part of melody from movie Titanic is given in Fig. 1.

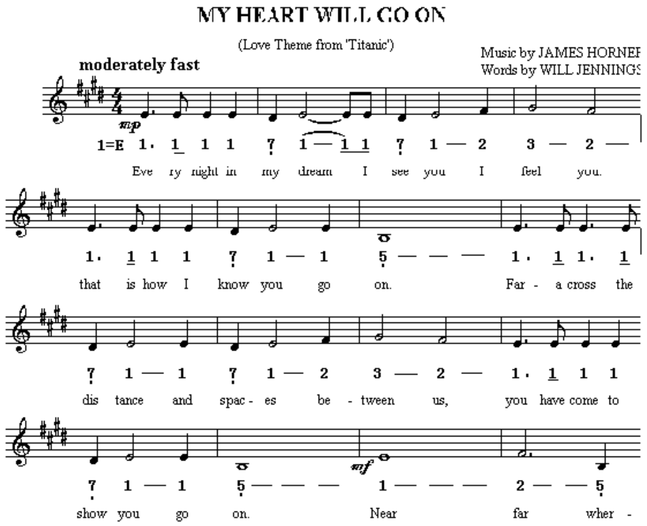

MY HEART WILL GO ON
(Love Theme from 'Titanic')
Music by JAMES HORNER
Words by WILL JENNINGS
moderately fast
mp
1=E
Eve ry night in my dream I see you I feel you.
that is how I know you go on. Far - a cross the
dis tance and spac - es be - tween us, you have come to
mf
show you go on. Near far wher -

Fig. 1. An example of stave notation and number notation

3 Melody Constraints

CDBs combine the constraint solving techniques with relational database theory, so that we can represent data in a more natural but more compact way, while the features of relational databases are still preserved. However, the constraint algebra should be redefined and designed according to the type of constraints. We have given the definition of CDB framework in Sect. 1, where constraints are reduced to *canonical form*: unique representation and no redundancy. Based on *canonical representation*, we can give the definition of operations such as *conjunction, disjunction, projection, negation,* etc.

We notice that, although the original constraints are variant, there could exist a unique representation as long as the primitive data represented by these constraints is fixed. It requires us to design such an algorithm that can always generate a unique representation from original formula for a fixed data set. There may exist many such algorithms. But as long as we use one from beginning to end, the *uniqueness* property is always preserved. Uniqueness property makes comparison of constraints possible and easier. For example, we can check whether two different constraints are equivalent or approximate to each other by comparing their forms.

A melody is mainly decided by notes. However, people can still recognize the melody if its notes are increased or decreased by same amount of interval, just it sounds a little higher or lower. The melody is fixed by the relative interval information in notes. This phenomenon is called *transposition invariant.* If we take notes as constraints, can we find melody's *canonical form*? Can we design such an algorithm that ensures us a unique representation for a fixed melody?

3.1 Pitch Constraints

Let x_i represent the pitch value, where $1 \leq i \leq m$, m is the number of pitches in a melody, and the binary operator "$-$" means the calculation of interval (the number of semitones) between two pitches.

Pitch's Local Constraint

The melody can still be recognized when its notes are increased (or decreased) by same amount of intervals, because the relative interval information between adjacent pitches is preserved. Therefore, given a melody, we have the following constraints:

$$x_{i+1} - x_i = N_i, \qquad 1 \leq i \leq m-1 \tag{1}$$

where N_i is a constant. Then a melody can be expressed as

$$\bigwedge_{i=1}^{m-1} (x_{i+1} - x_i = N_i) \tag{2}$$

Pitch's Global Constraint

From music theory, we have the following facts (see [1,8,15] also):

– Although one octave is divided into 12 notes, we only use a subset of them as a scale. For example, in C major scale C D E F G A B C, the intervals of E-F (the third and the fourth notes) and B-C (the seventh and the eighth notes) are semitones, and the others are wholetones. The *complement* set consists of 5 remaining notes.

- A piece of music is composed of notes in a scale and notes in the *complement* set. The music that consists of notes largely from a scale is called *diatonic* music, or *tonal* music, otherwise it is *atonal* music.
- Through thousands of years of musical evolution, most music, especially the popular music, are *diatonic* music.
- The notes in the complement set are considered unstable. Given a tonal context (music composed of a certain scale) , unstable element was itself poorly remembered [15]. For example, in C major scale, the C$^{\#}$ is hard for people to catch, remember, and sing.

Based on the above facts and observations, we make the following assumptions to simplify the problem: 1. people use notes in a scale evenly while composing music, or each note of a scale appears at least once in a melody. 2. do not use unstable notes, i.e., a melody does not contain unstable notes. Based on the above discussion and assumptions, we can summarize our results in the following Theorem:

Theorem 1 *All pitches from a melody form and follow a scale structure:*

$$\exists x_i \forall x_j (((x_j - x_i) mod 12) \in \{0, 2, 4, 5, 7, 9, 11\}) \tag{3}$$

which means that, we can find a note x_i which is the first element in a scale. The intervals (after mod) between x_i and other notes can only be the values in the set $\{0, 2, 4, 5, 7, 9, 11\}$.

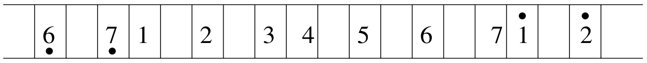


Fig. 2. The structure of a scale

Since the notes C D E etc. are absolute pitch names, denoting an absolute scale, we use numbers $1, 2, 3, 4, 5, 6, 7, \dot{1}$ to represent a general scale. One . on the top means one octave increase, and on bottom means one octave decrease. Fig. 2 is the structure of a scale (major/minor), where the vacancies are the unstable elements and can be written as 0.5 plus the previous number, e.g., C$^{\#}$= 1.5 in a C major scale.

```
Algorithm $Reduce$
    input: a sequence of pitches $p_1, p_2, \cdots, p_m$[3]
    output: a sequence of numbers $n_1, n_2, \cdots, n_m$
    let $Number[7][m]$ be an array to store number sequences
    let $Error[7]$ be an array to count the unstable elements
```

[3] p_i can be an absolute note such as C, D, etc., or the frequency value.

```
pick up $p_k$, the median from $p_1, \cdots, p_m$
for $i = 1$ to 7
    assign $p_k$ to location $i$ in Fig. 2
    for $j = 1$ to $m$
        assign location for $p_j$ according to the interval between
        $p_k$ and $p_j$ , and store the location in $Number[i][j]$
        if  $p_j$ is in vacancy
            $Error[i]++$
        endif
    endfor
endfor
return $Number[l]$, where $Error[l]$ is minimum $(= 0)$
```

Proof of Correctness:

Reduce returns the formula

$$\bigwedge_{i=1}^{m} (x_i = n_i) \tag{4}$$

where n_is are the number in the set {1,2,3,4,5,6,7} plus some ($\geq$ 0) dots on the top or bottom. Obviously, it satisfies formula (2), because $p_{i+1} - p_k = n_{i+1} - n_k, p_i - p_k = n_i - n_k \Longrightarrow n_{i+1} - n_i = p_{i+1} - p_i \Longrightarrow x_{i+1} - x_i = n_{i+1} - n_i = p_{i+1} - p_i = N_i$. For formula (3), we just need to choose the pitch which is assigned 1 as x_i, then the distance(after $mod12$) between x_i and any other pitch can be and only be in the set {0,2,4,5,7,9,11}, i.e., the global constraint is satisfied.

Proof of Uniqueness

Let $p_1, \cdots, p_m$ be a sequence of notes and p_k be the median note. Let Θ_i be an assignment with $i \in \{1, 2, 3, 4, 5, 6, 7\}$, p_k is assigned a location i in the scale structure and the relative intervals between any two notes are preserved. Let $\alpha(\Theta_i)$ be the number of notes that are not in vacancies, $\beta(\Theta_i)$ be the number of notes that are in vacancies.

Definition 1 *A perfect assignment Θ^{Δ} is the assignment Θ with $\beta(\Theta) = 0$.*

Definition 2 *By a reduced*[4] *scale we mean one octave only, and it consists of 7 elements, where the interval between the 3rd and 4th elements is one semitone, and the other intervals are one wholetone. A reduced assignment θ is defined as follows: modulate and assign notes into a reduced scale by shifting n octaves, where n is an integer.*

The five vacancies $\{1, 2, 3, 4, 5\}$ in one octave are written as 1.5 2.5 4.5 5.5 6.5 in ascending order of their positions. Given a melody and a reduced assignment

[4] Here the reduce is not the algorithm *Reduce*, but a mod operation over a data structure

θ, if we use ρ_j to denote the number of notes that are assigned j, and use ϱ_l to denote the number of notes that are in vacancy l ($j \in \{1,2,3,4,5,6,7\}$, $l \in \{1,2,3,4,5\}$), then we have:

Lemma 1 *Given a sequence of notes $p_1, \cdots, p_m$ and the median note p_k, we have $\alpha(\theta_i) = \alpha(\Theta_i)$, and $\beta(\theta_i) = \beta(\Theta_i)$, where $i \in$\{1, 2, 3, 4, 5, 6, 7, 1.5, 2.5, 4.5, 5.5, 6.5\}.*

Proof: After being shifted by n octaves, a note still keeps its original status: it is in vacancy iff it was in vacancy before shifting; it is not in vacancy iff it was not in vacancy before shifting. Therefore, the number of unstable elements is invariant under the assignment Θ_i and the reduced assignment θ_i.

Definition 3 *If a melody is composed of notes only from a scale, i.e., does not contain unstable elements, we call it a pure diatonic melody.*

Lemma 2 *For a pure diatonic melody, Θ^Δ exists.*

Proof: Let $1,2,3,4,5,6,7$ represent the elements in the reduced scale, then the reduced assignment θ ensures that the melody can be transformed into a sequence of numbers from the set $\{1,2,3,4,5,6,7\}$ only, i.e., $\beta(\theta) = 0$. Since $\beta(\Theta) = \beta(\theta) = 0$ (by Lemma 1), Θ^Δ exists.

Theorem 2 *Given a pure diatonic melody, if there exists such a reduced assignment θ that $\rho_j > 0$ and $\varrho_l = 0$, for $\forall j \in \{1,2,3,4,5,6,7\}$, $\forall l \in \{1,2,3,4,5\}$, then Θ^Δ is unique.*

Proof: From Lemma 2 we know that Θ^Δ exists, and the corresponding reduced assignment can be θ , since $\beta(\Theta) = \beta(\theta) = \Sigma\varrho_l = 0$. Let $C\ D\ E\ F\ G\ A\ B$ represent the original reduced scale, and let $\theta_1 = \theta$, i.e., θ is the assignment of $C = 1, D = 2, E = 3, F = 4, G = 5, A = 6, B = 7$. Then we will get the following assignments:

- θ_1: 1 2 3 4 5 6 7, $\beta(\theta_1) = 0$
- θ_2: 2 3 4.5 5 6 7 1.5, $\beta(\theta_2) = \rho_4 + \rho_1$
- θ_3: 3 4.5 5.5 6 7 1.5 2.5, $\beta(\theta_3) = \rho_4 + \rho_5 + \rho_1 + \varrho_2$
- θ_4: 4 5 6 6.5 1 2 3, $\beta(\theta_4) = \rho_7$
- θ_5: 5 6 7 1 2 3 4.5, $\beta(\theta_5) = \rho_4$
- θ_6: 6 7 1.5 2 3 4.5 5.5, $\beta(\theta_6) = \rho_1 + \rho_4 + \rho_5$
- θ_7: 7 1.5 2.5 3 4.5 5.5 6.5, $\beta(\theta_7) = \rho_1 + \rho_2 + \rho_4 + \rho_5 + \rho_6$

From the above, we know that there is only one perfect reduced assignment—θ_1, while $\rho_j > 0$, $\varrho_l = 0, \forall j \in \{1,2,3,4,5,6,7\}, \forall l \in \{1,2,3,4,5\}$. Therefore, Θ^Δ is also unique and equal to Θ_1 (Lemma 1).

Corollary 1 *If a pure-diatonic melody contains the fourth and the seventh elements of a reduced scale, Θ^Δ is unique.*

Proof: Obviously, when $\rho_4 > 0$ and $\rho_7 > 0$, $\beta(\theta_2), \cdots, \beta(\theta_7)$ are greater than 0. Θ_1 is the perfect assignment.

In practice, a piece of melody is composed of notes largely from a scale and a few from the vacancies, i.e., $\Sigma\rho_j >> \Sigma\varrho_l$. The following results can be derived:

- $\beta(\theta_1) = \varrho_1 + \varrho_2 + \varrho_3 + \varrho_4 + \varrho_5$
- $\beta(\theta_2) = \rho_1 + \rho_4 + \varrho_2 + \varrho_4 + \varrho_5$
- $\beta(\theta_3) = \rho_4 + \rho_5 + \rho_1 + \rho_2 + \varrho_5$
- $\beta(\theta_4) = \rho_7 + \varrho_3 + \varrho_4 + \varrho_1 + \varrho_2$
- $\beta(\theta_5) = \rho_4 + \varrho_4 + \varrho_5 + \varrho_1 + \varrho_2$
- $\beta(\theta_6) = \rho_1 + \rho_4 + \rho_5 + \varrho_5 + \varrho_2$
- $\beta(\theta_7) = \rho_1 + \rho_2 + \rho_4 + \rho_5 + \rho_6$

While considering the assignment on vacancies, we will have five more:

- $\beta(\theta_{1.5}) = \rho_2 + \rho_3 + \rho_5 + \rho_6 + \rho_7$
- $\beta(\theta_{2.5}) = \rho_3 + \rho_6 + \rho_7 + \varrho_3 + \varrho_1$
- $\beta(\theta_{4.5}) = \rho_5 + \rho_6 + \rho_1 + \rho_2 + \rho_3$
- $\beta(\theta_{5.5}) = \rho_6 + \rho_7 + \rho_2 + \rho_3 + \varrho_3$
- $\beta(\theta_{6.5}) = \rho_7 + \rho_3 + \varrho_1 + \varrho_3 + \varrho_4$

where Θ^{Δ} should be redefined to be the assignment with minimal $\beta(\theta)$.

Theorem 3 *If a melody is composed of notes satisfying:*

$$\begin{cases} \rho_1 + \rho_4 > \varrho_1 + \varrho_3 \\ \rho_4 + \rho_5 + \rho_1 + \rho_2 > \varrho_1 + \varrho_2 + \varrho_3 + \varrho_4 \\ \rho_7 > \varrho_5 \\ \rho_4 > \varrho_3 \\ \rho_1 + \rho_4 + \rho_5 > \varrho_1 + \varrho_3 + \varrho_4 \\ \rho_1 + \rho_2 + \rho_4 + \rho_5 + \rho_6 > \varrho_1 + \varrho_2 + \varrho_3 + \varrho_4 + \varrho_5 \\ \rho_2 + \rho_3 + \rho_5 + \rho_6 + \rho_7 > \varrho_1 + \varrho_2 + \varrho_3 + \varrho_4 + \varrho_5 \\ \rho_3 + \rho_6 + \rho_7 > \varrho_2 + \varrho_4 + \varrho_5 \\ \rho_5 + \rho_6 + \rho_1 + \rho_2 + \rho_3 > \varrho_1 + \varrho_2 + \varrho_3 + \varrho_4 + \varrho_5 \\ \rho_6 + \rho_7 + \rho_2 + \rho_3 > \varrho_1 + \varrho_2 + \varrho_4 + \varrho_5 \\ \rho_7 + \rho_3 > \varrho_2 + \varrho_5 \end{cases} \tag{5}$$

then Θ^{Δ} is unique.

Proof: The above constraints indicate $\beta(\theta_2) > \beta(\theta_1)$, $\beta(\theta_3) > \beta(\theta_1)$, $\beta(\theta_4) > \beta(\theta_1)$, $\beta(\theta_5) > \beta(\theta_1)$, $\beta(\theta_6) > \beta(\theta_1)$, $\beta(\theta_7) > \beta(\theta_1)$, $\beta(\theta_{1.5}) > \beta(\theta_1)$, $\beta(\theta_{2.5}) > \beta(\theta_1)$, $\beta(\theta_{4.5}) > \beta(\theta_1)$, $\beta(\theta_{5.5}) > \beta(\theta_1)$, $\beta(\theta_{6.5}) > \beta(\theta_1) \Longrightarrow \Theta_1$ is the unique perfect assignment.

Example 1 *Let's reduce pitch sequence E-E-F-G-B-F-E-D: we can take E as median note, if we assign E* 1*, then the result will be* $1 - 1 - 1.5 - 2.5 - 4.5 - 1.5 - 1 - 6.5$*, which contains 5 unstable elements. If we let E be* 3*, then we will get a perfect number sequence* $3 - 3 - 4 - 5 - 7 - 4 - 3 - \dot{2}$*, i.e., without any unstable element.*

Example 2 *In the theme "Love story", the notes' distribution on a reduced scale is $\rho_1 = 34$, $\rho_2 = 25$, $\rho_3 = 35$, $\rho_4 = 13$, $\rho_5 = 9$, $\rho_6 = 32$, $\rho_7 = 23$, and the unstable elements' distribution is $\varrho_1 = 1$, $\varrho_2 = 1$, $\varrho_3 = 1$, $\varrho_4 = 4$, $\varrho_5 = 0$. The perfect assignment is unique since the notes' distribution satisfies all the constraints in Theorem 3.*

Songs \ Assign p_k with			1	2	3	4	5	6	7
Name	p_k	m							
Happy birthday to you	G	25	7	0 △	4	16	2	2	7
Libiamo (from La Traviata)	D	292	165	72	23 △	176	132	60	40
Love story	E	178	81	52	7 △	113	57	19	30
Jingle bells	A	83	19	0 △	20	41	10	0 △	29
Red river valley	A	85	55	25	0 △	59	39	7	2

Fig. 3. Number of unstable elements $\beta(\Theta)$ under different assignment (Δ the perfect assignment)

Assumptions and Refinement of *Reduce*

The above proof is based on the assumptions: a melody keeps in one tonal context (doesn't change scale). However, the algorithm is still applicable in real practice, because most music, especially the popular songs, are tonal music and use notes largely from just one scale. The median note is usually located integer 1 to 7. However, we can refine the algorithm by iterating 12 times (or more) for each semitone in an octave (or more than one octave), in case the picked pitch p_k happens to be an unstable element.

Handling the Change of Tonal Context in a Song

The algorithm can also be used to tell whether a music changes its scale to another. For example, *Reduce* returns a number sequence with still very large number of unstable elements, which indicates that the melody probably changes the scale somewhere. Then the main tasks are to separate the input into two or more segments and *Reduce* them individually. We can solve the problem by auditing the density of error occurrences, and separate the sequence at the position where large number of errors begins . Most music use one scale as the tonal context. However, some music change scale once or twice in order to express different emotions, e.g., at the end of the theme from Titanic, the original scale is increased by 5 semitones.

More Than One Perfect Representation

We notice that the choice of notes in composing a song may not be even, e.g., some music may use 5 notes only in a scale. Therefore, *Reduce* algorithm may generate more than one perfect sequence (see *Jingle bells* in Fig. 3), especially when the fourth or the seventh note of the scale is not used in the melody. Let's illustrate by example "Beethoven's Ninth Symphony, Movement 4": the input pitches can be {EEFG GFED CCDE EDD, EEFG GFED CCDE DCC, DDEC DEFEC DEFED CD$\underset{\cdot}{G}$, EEFG GFED CCDE DCC}. Since there is no A and B in the above, we will have two perfect representations {3345 5432 1123 322, 3345 5432 1123 211, 2231 23431 23432 12$\underset{\cdot}{5}$, 3345 5432 1123 211}, and {77$\dot{1}\dot{2}$ $\dot{2}\dot{1}$76 5567 766, 77$\dot{1}\dot{2}$ $\dot{2}\dot{1}$76 5567 655, 6675 67$\dot{1}$75 67$\dot{1}$76 562, 77$\dot{1}\dot{2}$ $\dot{2}\dot{1}$76 5567 655}.

To guarantee uniqueness, we need another algorithm *Weight*: For each scale note, we assign a weight p according to its probability of being used. For example, the most frequently used notes are 1 to 6. So we can assign higher weight to 1 to 6, lower weight to $\underset{\cdot}{5}7\dot{1}\dot{2}$. Then we can scan the two perfect sequences, and sum their weight respectively. The sequence with higher weight is our choice. In other words, we prefer the one whose notes are more centrally located in a scale (in the above example, $\{3345\ 5432\ 11 \cdots\}$ is better than $\{77\dot{1}\dot{2}\ \dot{2}\dot{1}76\ 55 \cdots\}$). The *Reduce* and *Weight* algorithms are both $O(n)$, and can be combined into one, e.g., assign unstable element -1, and stable element a positive value according to its frequency of usage. More precisely, we can take duration of the pitch into account, i.e., $Weight = \sum_{i=1}^{m} p_i \times y_i$, where p_i is the preference of using this note in a scale, and y_i is its current duration.

$$Max(\sum_{i=1}^{m} p_i \times y_i) \tag{6}$$

(6) is a more precise description of melody's global constraint than (3), where the usage preference of each number needs further investigation. *Weight* returns a most stable assignment, which has less unstable elements and the notes are distributed centrally on a scale.

3.2 Duration Constraints

Let y_i be the duration of x_i, where $1 \leq i \leq m$.

Duration's Local Constraint

We know that, if we change the melody's rhythm, it just sounds faster or slower than the original. It is easy to find the following constraint between two adjacent pitches:

$$\frac{y_{i+1}}{y_i} = D_i, \qquad 1 \leq i \leq m-1 \tag{7}$$

where D_i is a constant. The tuple can be written as:

$$\bigwedge_{i=1}^{m-1} (\frac{y_{i+1}}{y_i} = D_i) \tag{8}$$

Actually people also have a general preference on durations, which means we can define unstable elements for a duration structure similar to defining unstable elements in an octave. Then the uniqueness of the representation can be guaranteed. The preference of duration structure is not as obvious as that of a scale, and needs further investigation.

Example 3 *The sequences of durations (measured in second)* $\{\frac{1}{4}, \frac{1}{2}, \frac{1}{4}, \frac{3}{4}, 2, 1\}$ *and* $\{\frac{1}{2}, 1, \frac{1}{2}, \frac{3}{2}, 4, 2\}$ *are identified the same, and Reduce returns* $\{1, 2, 1, 3, 8, 4\}$ *for them.*

Adding duration factor into the representation, we get a two-dimensional description of a melody, $\{(n_1, d_1), (n_2, d_2), \cdots, (n_m, d_m)\}$, which is unique in general.

3.3 Conclusion

Melody consists of two classes of constraints: local constraints and global constraints. Obviously the new representation is more compact, while the correctness and uniqueness are guaranteed. The major operation in melody retrieval is the substring matching, which we can adapt techniques from bioinformatics.

Music is one of the most beautiful and also complex phenomenon in our world. A delicate model of the melody should capture diverse features. Besides the crucial factors in the melody, our model captures the principle of diatonic music. We prove that it is applicable in general, i.e., we have a unique representation for a fixed melody.

From psychological point of view, we can explain our model as follows: through thousands of years of music development, people have built up a general preference or rules to create and appreciate the music. Harmonic music is usually taken as that obeys these rules. Although notes in an octave are equally spaced into 12 parts, in fact people don't treat them equally. Some notes are taken as unstable elements. Our algorithm *Reduce* is to find an assignment with the least unstable elements, which is very similar to people's cognition procedure, "... the unstable element was itself poorly remembered" [3].

4 Experimental Results

Our Database is a collection (more than 100) of popular songs from different countries, and a short piece (consisting of k notes) of melody can be looked as a query. First, after analyzing scores made by musicians we figure out a rough preference about scale notes (Fig. 4).

Our algorithm *Reduce* and *Weight* aim at generating a unique representation for a piece of melody. The main concern is whether or not this representation is the same as that translated by musicians, which we call the correctness of our translation. In the following tables, we consider these algorithms:

– *Reduce*: generates melody's representation by counting the number of unstable elements.

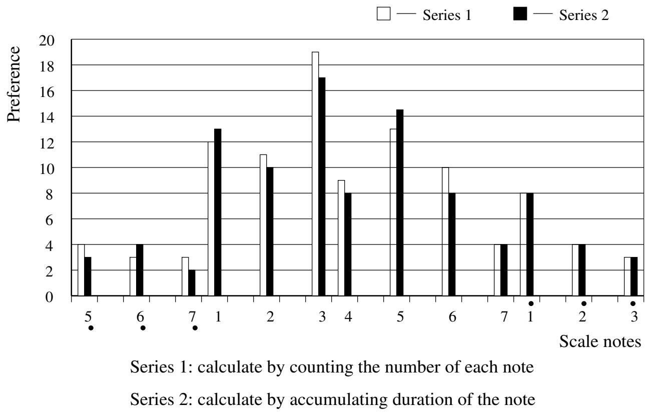


Fig. 4. The preference of scale notes

- *Weight*: based on the general note-preference table (Fig. 4), returns the most stable representation.
- *Reduce+Weight*: use *Reduce* to generate some candidates first, then choose the most stable one from these candidates by *Weight*.
- *The second candidate*: if the first candidate returned by *Reduce+Weight* is wrong, then we consider the second candidate as the correct representation. The experiments show that, after choosing the second candidate, correctness can be guaranteed.

Table 1: Correctness of unique representation for songs

Reduce	***Weight***	***Reduce+Weight***
61.2%	80.6%	100.0%

Table 2: Correctness of unique representation for queries

Alg. ***k***	***Reduce***	***Weight***	***Reduce + Weight***	***The second candidate***
k>3	13.3%	59.6%	76.6%	99.5%
k>5	14.4%	61.2%	78.7%	100.0%
k>7	17.9%	62.9%	81.4%	100.0%
k>9	22.1%	63.5%	83.9%	100.0%
k>11	32.5%	65.5%	87.5%	100.0%
k>13	38.5%	65.5%	96.2%	100.0%

Fig. 5. The correctness of translation algorithms

Table 1, the correctness for translating songs, indicates that the correctness can be guaranteed if we combine *Reduce* and *Weight*. Table 2 is for the translation of query which is much harder to get a unique and correct representation. We find that, the longer the query is, the better the algorithm performs. It's better to let $k > 7$. The duration representation can be found in the similar way, just need to figure out the unstable duration elements.

5 Application and Future Work

For last several decades content-based retrieval has been explored in many different forms of media. Large volumes of music are now available and stored in MIDI or MP3 format. A nature way of querying songs is by humming or singing a piece of melody. Much research [6] of extracting pitches from MIDI or other audio files have been done in recent years, and pitches representing the main melody in polyphonic music can be retrieved and handled also [4,5,7].

In current music retrieval systems, such as SEMEX, MELDEX, Pollastri(99), the melody is represented by contour-based string. Due to the property of transposition invariant, storing and comparing absolute pitches is meaningless, but a coarser melodic contour description is more important to listeners in determining similarity. The U D S is a 3-level contour representation. Similarly, 5-level and 7-level contour representations [19] have better performance in searching, but need to store more information. Another approach is to store the intervals between pitches [13], e.g., transform C D F D D A into 2 5 -2 0 7. The query becomes the substring matching problem [2,6], which we can adapt techniques from bioinformatics—DNA and protein sequence matching. In [12], suffix-trie is proposed to store and index music data.

However, the above two representations didn't consider global constraints. For example, {FGAB} and {CDFB} can only be translated into {4567} and {1247} respectively, by our method. But their contour-based strings are same, i.e., UUUU.

The interval-based representation is not efficient in searching, since it will have many wrong entries, for example, {CDECDEFB...} and {FGAFGABG...} are different melodies but have the same intervals for first six notes, 2 2 -4 2 2. The system can not tell the difference between two melodies on the first 6 notes. But our algorithms generate different sequences, {12312347} and {45645675} respectively, and enable us to distinguish them immediately. Since we consider the global constraint in pitches, a melody is translated into more specialized string. Because the global information has been encoded into every number, we can tell that the two pieces of melody are under very different contexts while comparing their first numbers 1 and 4, i.e., the first pitches(1,4) take the roles of the first element and fourth element in the scale, respectively. Therefore, though the two sequences have similar contours, they should not be considered as a match.

In contrast to contour-based and interval-based methods, our constraint-based representation takes *global constraints* into account, based on the fact that the pitches chosen to compose harmonic music are under a *tonal* (global) constraint. In other words, contour-based and interval-based approaches only consider the relation between two adjacent pitches, while our constraint-based approach considers the relation among all the pitches in a melody. By calculating canonical form of melody constraints, we can represent music simply, without losing preciseness and query efficiency. This representation also provides us a tool for further investigation on music. For example, the key numbers (the elements in the scale that a melody focuses on) of a melody can be easily figured out by their usage frequency and the song's ending note. We find that the music with major tune usually focuses on 1 3 5, while minor tune on 2 4 6. Classic Chinese music only uses 5 notes as a scale. Classic Russian music contains many $4^{\#}$ and $5^{\#}$.

The third approach is to transform melody into time-series data[20], where the notes are reduced by subtracting the average of the piece. However, the average varies, especially when some notes in the piece are missing or the length of piece changes. On the contrary, our approach can handle the above problems very well: as long as the context hasn't been changed (usually means a different melody if it changes), the change on the piece will not affect the translation of the remaining notes.

The future work includes: 1. detecting the change of tonal context in a song, and separate the input properly. 2. define a duration structure (similar to the scale structure) and make a unique 2-dimension representation for melodies. 3. design a new query mechanism for music data: instead of the traditional method, dynamic programming, we are considering a memory-reduced approach. Taking full advantages of 2-dimension representation, the memory-reduced approach is expected to improve the query process on both efficiency and accuracy.

6 Conclusion

The idea in this paper is somehow similar to some pitch spelling algorithms. For instance, Longuet-Higgins's algorithm [14] computed the value of "sharpness" for each input note, then tried to spell notes so that they are as close as possible to the local tonic on the stave notation. In this paper, we not only provide a much simpler processing algorithm by symbolizing notes with proper numbers and applying constraint solving techniques, but also prove that uniqueness of this representation can be guaranteed generally, even for a short piece of the whole melody. Additionally, we discuss some issues in the application of this method in melody retrieval, and propose several feasible solutions.

To make a more delicate model to describe melody, we need to consider some other factors: timbre, beat, stress of pitches, etc. We expect to extend our model with these features and build a practical and efficient music database system. Tonal context is only one of the constraints in popular music. Our future investigation also includes using mathematical modelling approach to study other constraints in harmonic context.

References

1. Alexandra. L. Uitdenbogerd, Justin Zobel. Manipulation of Music for Melody Matching. ACM Multimedia 98 Proceedings, pp. 235-240. 1998.
2. Alexandra. L. Uitdenbogerd, Justin Zobel. Matching Techniques for Large Music Databases. Proceedings of the Seventh ACM International Multimedia Conference, pp. 57-66. November, 1999.
3. C.L. Krumhansl. Cognitive Foundations of Musical Pitch. New York: Oxford Unversity Press. 1990.
4. Costas S. Iliopoulos, Kjell Lemstrom, Mohammed Niyad, Yoan J. Pinzon. Evolution of Musical Motifs in Polyphonic Passages. Proc. AISB'2002 Symposium on AI and Creativity in Arts and Science, pp. 67 - 75. April 2-5, 2002.
5. Dave Meredith, Geraint A Wiggins, Kjell Lemstrom. Pattern Induction and Matching in Polyphonic Music and other Multidimensional Datasets. Proc. the 5th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI2001), Volume X, pp. 61 - 66. July, 2001
6. David Bainbrige, Craig G. Nevill-Manning, Ian H. Witten, Lloyd A. Smith, and Rodger J. McNab. Towards a Digital Library of Popular Music. Proceedings of the fourth ACM conference on digital libraries, pp. 161-169. 1999.
7. Geraint A. Wiggins, Kjell Lemstrom, David Meredith. SIA(MA)ESE: An Algorithm for Transpostion Invariant, Polyphonic Content-Based Music Retrieval. 3rd International Symposium on Music Information Retrieval. 2002.
8. Ilya Shmulevich, Olli Yli-Harja, Edward Coyle, Dirk-Jan Povel, Kjell Lemstrom. Perceptual Issues in Music Pattern Recognition—Complexity of Rhythm and Key Finding. Proc. AISB'99 Symposium on Musical Creativity, pp. 64-69. April, 1999.
9. J.D. Ullman Principles of Database Systems. Computer Science Press. 1983.
10. Kim Marriott, Peter J. Stuckey. Programming with Constraints: an Introduction. MIT Press, 1998.
11. Kjell Lemstrom, Pauli Laine, Sami Perttu. Using Relative Interval Slope in Music Information Retrieval. Proc. 1999 International Computer Music Conference (ICMC '99), pp. 317-320. October 23-29, 1999.
12. Kjell Lemstrom, Atso Haapaniemi, Esko Ukkonen. Retrieving Music—To Index or not to Index. Proc. Art Demos - Technical Demos - Poster Papers - The Sixth ACM International Multimedia Conference (MM '98) pp. 64-65 + loose sheet. September, 1998.
13. Kjell Lemstrom, Pauli Laine. Musical Information Retrieval Using Musical Parameters. Proceedings of the 1998 International Computer Music Conference. Ann Arbor, MI, pp. 341-348. 1998.
14. Longuet-Higgins, H.C. The Perception of Melodies. In S.M. Schwanauer and D.A. Levitt, editors, Machine Models of Music, pages 471-495. M.I.T. Press, Cambridge, Mass. 1993.
15. Olli Yli-Harja, Ilya Shmulevich, Kjell Lemstrom. Graph-based Smoothing of Class Data With Applications in Musical Key Finding. Proc. IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing, pp. 311-315. June, 1999.
16. Paris C. Kanellakis, Gabriel Kuper, and Peter Z. Revesz. Constraint Query Languages. Journal of Computer and System Sciences, 51(1):26-52. August 1995.
17. Roger B. Dannenberg. Music Representation Issues, Techniques,and Systems. Computer Music Journal, 17(3), pp. 20-30. 1993.
18. Volker Gaede, Mark Wallace. An Informal Introduction to Constraint Database Systems. Constraint Databases and Applications pp.7-52. 1997.

19. Youngmoo E. Kim, Wei Chai, Ricardo Garcia, Barry Vercoe. Analysis of a Contour-Based Representation for Melody. Proceedings of International Symposium on Music Information Retrieval. October, 2000.
20. Yunyue Zhu, Dennis Shasha. Warping Indexes with Envelope Transforms for Query by Humming. SIGMOD Conference: 181-192. 2003.

Music Segmentation: An XML-oriented Approach

Goffredo Haus, Luca A. Ludovico

LIM-DICO University of Milan
Via Comelico, 39
20135 Milano, ITALY
`haus@dico.unimi.it, luca.ludovico@dsi.unimi.it`

Abstract. In this paper, two related subjects are discussed: musical segmentation and the representation of its results in a particular XML encoding, namely MX. About segmentation, there is a comprehensive discussion of the main ideas and the employed operators to implement it. Besides, in this paper we describe the principles of our music XML format.

Introduction

This paper represents the results recently obtained at LIM (Musical Informatics Laboratory, State University of Milan) in the area of musical computer science. In particular, the document is focused on the automatic analysis of musical works in order to recognize and extract musical objects. Our purpose is to encode both music information and corresponding meta-data in a unique data structure. From this point of view, XML provides an effective way to represent musical information at different levels of abstraction. In the format we propose, namely MX, it is possible to correlate notational symbols as well as audio fragments, and printed scores as well as performance files. Our encoding format is particularly suitable to represent information coming from a manual or automatic segmentation process. Thanks to its multi-layer structure, themes and other musical objects can be referred not only to organized symbols in score, but also to audio performances and printed documents. XML encoding and segmentation will be the main subjects of the following discussion.

XML and Music Information

XML provides indeed an innovative way to represent information. The purpose of our efforts is to benefit by XML in order to generate a complete description of musical communication. From this point of view, XML presents a number of remarkable features:

- **Intelligibility:** compared to other encoding formats, a markup language is more intelligible. Thanks to the tag mechanism, the receiver of music communication (both a computer user and a musician) can easily retrieve information. A specific software application is not required, as information is encoded in plain text format.

- **Implementability:** musical applications using XML are implementable. A number of XML-based software were developed, both to represent and to manipulate music information. Possible manipulations include not only simple editing but also meta-data extraction and segmentation.
- **Extensibility:** an XML format can be extended to represent data, meta-data and layers previously ignored. Dealing with a dynamic field such as music production, we can not neglect possible future evolutions: for example, new notational symbols in score or new levels of abstraction in multimedia communication.
- **Hierarchical structure:** music information is strongly structured. A music piece is made of one or more pages, each one containing one or more staff groups, each one containing one or more staves and so on… (parts on a staff, voices in a part, beats for a voice, chords in a beat, notes in a chord). This example shows a hierarchical structure, which can be reflected and properly represented by XML.

Thus, XML is an effective way to describe music information. Nowadays, there is a number of good dialects to encode music by means of XML: MusicXML, MusiXML, MusiCat, MEI, MDL etc. (see [11] for a thorough discussion). In particular, we have at least two good reasons to mention MusicXML by Michael Good – Recordare. First, it can be considered a good and comprehensive way to represent symbolic information. As a consequence, MusicXML was integrated in a number of commercial programs. Among them, it's worthwhile to cite one of the leading applications for music notation: Coda Music *Finale*. And another advantage of MusicXML is represented just by its popularity in the field of music software.

We developed a new XML-based format, called MX. Our approach is different from the aforementioned ones thanks to the following key features: the *multi-layer structure* for music information and the concept of *space-time construct*.

In our opinion, musical information can be (and should be) structured by using a layer subdivision model, as shown in Fig. 1. Each layer is specific to a different degree of abstraction in music information. In our proposal for a common and exhaustive format, we distinguish among General, Structural, Music Logic, Graphic/Notational, Performance and Audio layers (see Figure 1a). For example, MusicXML could be integrated in the more comprehensive MX encoding to implement the Logical Organized Symbols layer, that is symbolic information in score (notes, rests, articulations,…); whereas other common file types can be linked to represent other layers: TIFF for the Notational layer, MP3 and WAV for the Audio layer and so on. There is a good coverage of the matter in [11].

Considering music structure as multi-layered, we need a sort of glue to keep together the heterogeneous contributions. Accordingly, we introduced the concept of spine. *Spine* is a structure that relates time and spatial information (see Figure 2), where measurement units are expressed in relative format. Through such a mapping, it is possible to fix some point in a layer instance (e.g. Notational) and search the corresponding point in another one (e.g. Performance or Audio). Later in this document, a complete example is provided.

Currently, MX is undergoing the IEEE standardization process, as described in [1]: Recommended Practice for the "Definition of a Commonly Acceptable Musical Application Using the XML Language" (IEEE SA 1599, PAR approval date 09/27/2001).

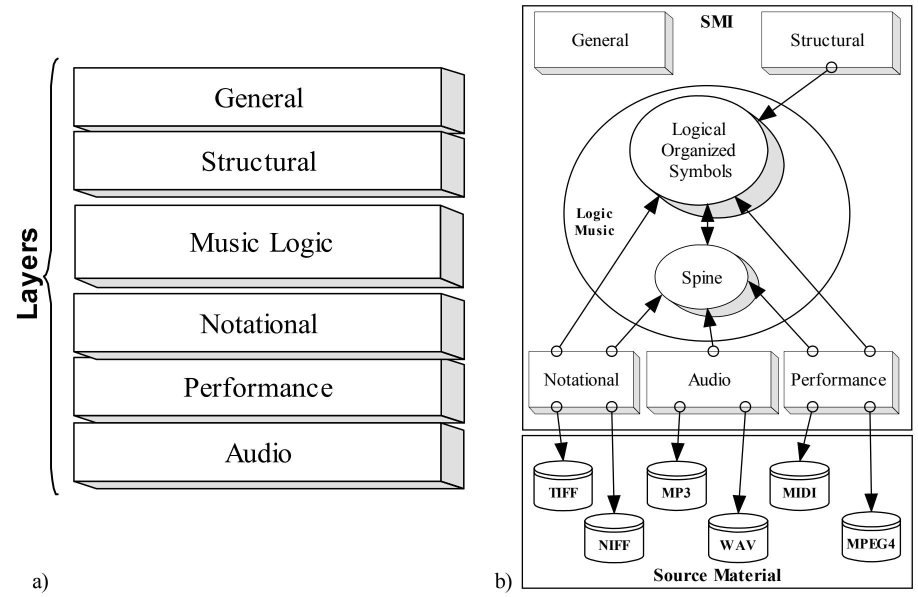


Fig. 1 – (a) Music information layers and (b) relations among them

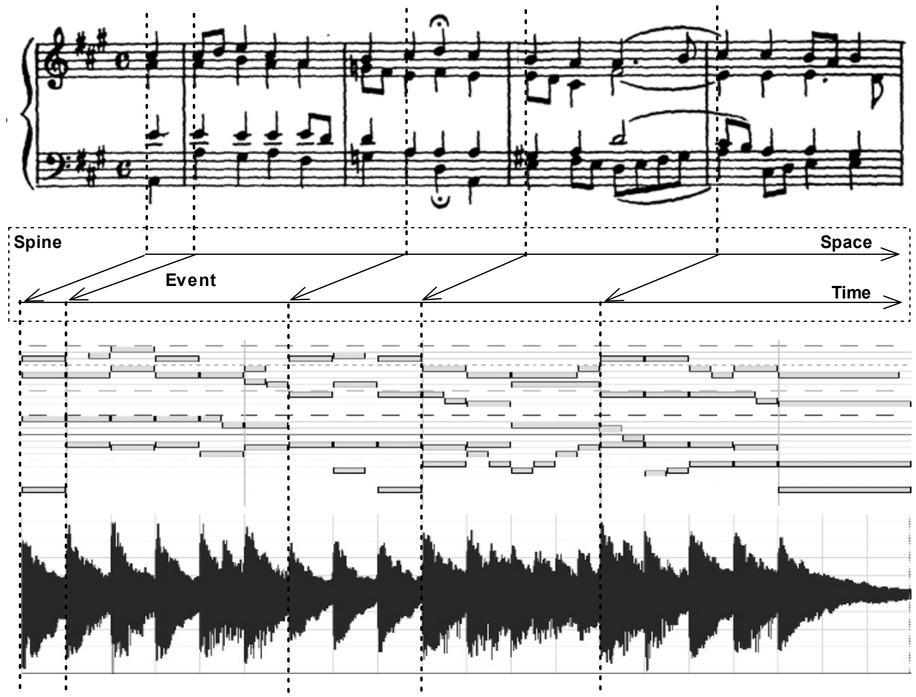


Fig. 2 – Spine: relationships between Notational and Performance/Audio layer

A Brief Comparison Among MX and Other Formats

As mentioned before, we know that some of the basic concepts related to MX could be considered not completely original. As an example, we cited other XML-based formats to encode music information, such as MusicXML and MEI. As regards this matter, MX is original thanks to its richness of contents. In fact, most XML-based codings aim at the representation of only one layer or a small subset of the layers listed before, whereas MX tries to depict music in all its aspects, logical as well as structural, aural as well as graphical.

Another interesting format to describe multimedia information is MPEG-7 , formally named "Multimedia Content Description Interface". Among its noteworthy features, the following are particularly close to our goals:

- The possibility to support a wide range of formats and file types: still pictures, graphics, 3D models, audio, speech, video, and composite information
- The presence of audiovisual description tools, pointing out the metadata elements, their structure and relationships in order to create descriptions which should allow effective and efficient access to multimedia content.
- The fact that the provided description is independent on the ways the described content is coded or stored.

However, MX purposes substantially differ from the MPEG-7 ones. In fact, the latter considers general multimedia information, whereas MX supports multimedia aspects of musical information. In other words, our format doesn't integrate graphics, audio or video elements as a part of a multimedia presentation, but as peculiar descriptions of music content. In this sense, we can say that MX is more specific about musical field, and music score (in its different aspects) remains the central point of our view. Besides, MX provides all the advantages of a plain XML coding we listed in the beginning of this paper. Providing a format easy to be read, modified and saved even by very common software tools is considered a peculiar aspect of our efforts, above all in a moment when the scientific and technological community is divided about problems such as network sharing, copyright, and open source solutions.

Segmentation

After a discussion about the key aspects of MX, let's consider the second matter of the paper. Segmentation implies recognizing and extracting main musical contents. This evaluation can be performed involving either a score or an audio recording. The two approaches are usually referred to as *notational* and *audio segmentation*, and are wrongly considered independent. Studies on relationships between both representations are currently in progress: the purpose is to reduce the distance between the former and the latter, proposing integrated-analysis techniques on audio and notational layer.

The target of segmentation is, at first, the aggregation of significant groups of elements: melody, rhythm or harmony patterns, eventually combined. The next step is the analysis and extraction of music objects, i.e. those parts of a piece characterized by common features. The meaning of the locution "common features" is too general, but soon it will become clear. As mentioned before, musical language is very rich and complex, and information can be conveyed at different levels of abstraction. So, an example of segmentation at notational level could be: “Let’s consider all the notes carrying a staccato articulation symbol”. This could represent a form of segmentation, but really poor from a musicological point of view. Another type of interrogation could be the following: “Let’s take into consideration all the sounds emitted by horns in an orchestra score”. Now segmentation could occur either at audio or at notational level, and would have its own musical meaning, indeed. But in the former case it would be a very hard task to extract the horn part from an orchestral tessitura;[1] on the contrary, considering notation, the problem would become even trivial. The question is: among all the possible contents we can search for in music, what is really relevant? What do we expect to determine?

Our approach is trying to partition music in order to recognize themes and other important recurrent musical objects. Our search is oriented toward generative material extraction from scores, including notation, phrasing and, at last, semantics. And we would like to reach this goal by the automatic process of segmentation we will briefly introduce in the following sections.

This computer-based process requires to pay a great attention to the correct evaluation of results. For instance, a strongly repetitive sequence of notes (pitches and rhythmic values) could highlight an important musical figure, such as a theme, as well as an irrelevant accompaniment.

In the structure depicted in Figure 1b, the starting point of our computations is represented by score or performance file formats, such as NIFF, MIDI, and MX itself. These files are parsed by our application software, namely Theme Finder, and transformed into a common notational encoding. Then, analysis process can begin and the corresponding results are finally written into an MX file. In the next section, our segmentation process and employed techniques will be detailed.

The Concept of Musical Object

We can refer to a “musical object” as every type of structured information in any musical language. As mentioned before, our purpose is the extraction of thematic elements. Themes and musical objects are formalizations of general linguistic concepts, thus the risk is to introduce sensitive restrictions based on aesthetic, historical and formal criteria.

Even at basic symbolic level, musical language has a multidimensional syntax, where an interweaving of rhythm, melody and harmony can be found. The musical alphabet is composed by a combination of at least two data for each symbol. The most atomic sound element, called the *note*, joins a sound pitch (frequency) to a rhythmic value (time duration). Usually, musical alphabet is more complicated: for instance, at rhythmic level not only durations but also accents play an important role. And, in order to have a more complete characterization of elementary information, we should also consider the loudness of note sound, and the timbre of the playing instrument .

[1] This is the problem usually referred to as *demixing*. Many studied about this matter are currently in progress, also at LIM.

However, a simplification of the model by considering only rhythm, melody and harmony dimensions can be sufficient. In fact, mental interpretation process doesn't force us to distinguish different instruments (even if it actually does): for instance, music themes are recognized even if they are split on different voices. And loudness is never related to a single language particle but to more complex expressive structures .

Each melodic line, or real voice, is made of notes and is decomposable in two dimensions immediately perceived by the listener. These dimensions are the rhythmical-melodic character (horizontal dimension) and the harmonic character (vertical dimension). The latter aspect is not referred to a single melody, but has to be evaluated together with other contemporaneous melodic lines. This originates the almost independent dimension of tonal functions.

The dimensional complexity just described, even if reduced and simplified, makes musical language harder to be formalized than other natural languages. Besides, only a few elementary harmonic patterns (such as tonic-subdominant-dominant-tonic), some melodic movements (e.g. sensible-root), and some rhythmical punctuation features (pauses, long-duration notes,…) apparently keep a unique definition or a commonly accepted meaning in all historical ages and cultures.

The main problem we have to deal with is the fact that a musical object is recognized also in presence of a slightly different information flow. Variation (in the form of embellishment, transposition, retrogradation, and so on…) is probably the most important method used by composers to develop music contents, and it can not be ignored. Thus, our pattern matching techniques not only will take into consideration literal repetitions, but also a number of conceivable variations and adjustments.

Of course, automatic segmentation is a difficult task to perform. At the moment, human intervention, even if not required, is still desirable. The supposed music themes must undergo a hand-made musicological evaluation, finalized to recognize their expected relevance and their completeness. An automatic process could extract a musical theme which is too long, or too short, or simply insignificant. That’s why a human feedback is still required in order to obtain high-quality results.

Introduction to Musical Operators

Now it should be clear that our goal is the automatic recognition of musical objects, finalized to their representation in a commonly accepted standard format.

In order to find out musical objects through a computer system, a number of musical operators must be defined. Musical operators are an algorithmic tool-set able to identify and report various types of musical objects.

Experience shows that limiting analysis to a single layer would produce a poorly meaningful segmentation. That’s why a great effort was done to design and implement data structures and algorithms suitable to multi-layer analysis. In particular, the newest version of our software, Theme Finder 3.0, is able to recognize and automatically extract musical objects using overall information in an integrated way.

As mentioned before, we consider essentially three layers: melody, rhythm and harmony. To each layer corresponds a subset of specific operators. The melodic, rhythmic, and harmonic operators we conceived are able to produce compatible and complementary results, even if they are very different in meaning and operating fashion.

Keeping the computational complexity fair should be one of the purpose of any implementation. To achieve this goal, our integrated analysis employs a "nearly diagonal" approach: computations on different layers occur in different moments, but the overall outcome is produced using all available information. Besides, during the analysis process, the extracted information “flows” among the levels, thus working as a guide for the following musical operators.

Unfortunately, in this short paper it is impossible to list all the operators with their algorithms, functions and structures. [7] and [12] cover this topic in detail. Here we will describe only their general principles and their main functionalities.

Redundancies and variated repetitions are recognized by applying a pattern matching technique. In particular, most operators consider two given sequences of notes, each one containing *n* symbols, and calculate a vector for each sequence. Of course, the contents of the vector depend on the type of operator being applied. For instance, a melodic operator is interested in pitch evaluation; thus, the vector elements will contain numbers that mathematically describe the pitch information. Then, the contents of the vectors are compared, and once again the way of matching is due to the specific operator. As a result, an error vector is generated. In a certain sense, the error vector measures the distance between two candidate musical objects. If all its components are null, then there is a perfect matching (according to that particular operator); if not, we must introduce a less rigid metrics than a simple binary “matched/unmatched” logic. To evaluate properly varied recurrences, very common phenomena in music compositions, a concept of tolerance must be introduced in the process of analysis.

Rhythmic/Structural Operators

This first class of operators focuses on rhythmic analysis. As a matter of fact, rhythmic patterns can be considered a way to describe musical objects.

When talking about rhythm, usually we think about durations. Of course, note (and rest) lengths represent an important aspect of rhythm. As a consequence, one of the key features of our operator is the length matching.

But there is another aspect we must deal with: the accent arrangement. The matching between two rhythmic objects cannot ignore also the congruence of accents. An accent is a “conceptual reference” of a perceptive feature; it can explain the “rhythmic tension” a note assumes due to its position in bar.

Notes can be grouped into equivalence classes, depending on note position in bar, beat (strong time, stasis, pulse) and upbeat (weak time, impulse) of bar, regular or irregular groups, syncopations and so on… Starting from score information, it is possible to use a simple automaton to assign rhythmic accents to each note in a melody. A possible implementation is discussed in [7].

Before performing an automatic comparison of fragments, our operator weights how much the position in bar is discriminating for the algorithm.

The parsing of rhythmic dimension has the advantage of being performed in a linear time cost complexity. Furthermore, the mathematical model is isomorphic to relative musical theory, because music division is based on mathematical rules. Thus, we can affirm that the theory is consistent.

In conclusion, two fragments will be considered completely equivalent from a rhythmic point of view only if both duration and accent equivalence is found. In our implementation, these two comparisons are unified under a single rhythmic/structural operator.

Melodic Operators

Also melody can be investigated by using pattern matching techniques. In this case, variations and their relationships with the original fragment have a particular importance. The way musical sequences are revised ranges from a simple reintroduction of the original phrase in a new tonality to complex counterpoint devices.

Thanks to the melodic operators we introduced, we are able to recognize musical modifications of four types:

- Transposition (real and tonal)
- Inversion (real and tonal)
- Retrogradation
- any combination of the preceding operators.

Transposition implies moving a note or collection of notes up or down in pitch by a constant interval (real transposition) or by a constant number of grades (tonal transposition). The *inversion* of a given melody is the melody turned upside-down. For instance, where the original melody has a rising third, the inverted melody presents a falling third; once again, intervals can undergo a real inversion (e.g., a rising major third becomes a falling major third) or a tonal one (e.g., instead of moving two grades up, melody falls two grades down). *Retrogradation* is the movement in the opposite sense: the last note of the original sequence becomes the first note in the varied one, and so on…

In order to apply melodic operators, a formal model of musical scale must be provided. As mentioned before, there are two distinct ways to measure pitches, and therefore to evaluate distance between notes:

- a method based on chromatic distance, i.e. on half-tones; this technique is insensitive to the tonal context.
- another based on diatonic distance, i.e. on grades of a scale; this method considers the actual tonality.

In order to implement both real and tonal operators, we need both the enumerative systems: one to calculate the real distance, another to calculate the tonal distance (see [4] and [12] for a detailed discussion).

Melody is a sequence of pitches, comparison is performed on this succession. Let's notice that there is a perfect informative equivalence between a description of absolute pitches and a sequence of melodic interval, where only the first note is eventually described by its absolute pitch whereas the following are identified by the distance from the preceding one. The latter represents a differential approach, the one we choose in our implementation for the following reasons:

1. Human ear naturally evaluates pitch variation between notes instead of absolute frequency of each note. Most listeners recognize a melodic pattern by matching pitch sequential differences (as well as rhythm, of course).
2. In our implementation, absolute values are not important to determine recurrences and variations. This approach allows to code only differential information, saving hardware and computational resources: for instance, a *n*-notes sequence can be represented by a (*n*-1)-items vector.

Harmonic Operators

Harmonic approach completed our analysis system at semantic-contextual level. Melodic and rhythmic/structural operators highlight good local solutions, but they can not provide a structural cognition of the musical meta-language ruling phraseological generation. The aforementioned operators can be used for a syntactic analysis of musical language, but harmony evaluation is a powerful instrument to identify sections and phrases within the composition plan. In this sense, harmony analysis is a helpful cognitive tool for understanding phrase construction and for limiting its scope. Compared to melodic and rhythmic/structural operators, harmonic operators work at a higher level of abstraction, while discovering a deeper structure.

The following operators work serially in order to perform the desired analysis.

The first harmonic operator is *Verticalization*, used to identify the temporal occurrence of a sound event. It is essentially a "selective" function, with a domain defined by all the notes in the score. This function groups notes in chords. Verticalization operator is able to distinguish and manage redundant sounds that harmonic analysis should not consider.

The second harmonic operator is called *Tonal Function Recognition*. This operator evaluates the chord passed as argument, and return a symbol of musical meta-language [17]. Its purpose is not only of recognizing the grade of the scale where the chord is built, but also of providing a musical explanation of the chord function. The final goal of TFR operator is the reduction of a whole set of notes (corresponding to single chords) in a symbol belonging to one of three classes: Tonic, Dominant or Subdominant. These classes correspond to the key harmonic regions in a musical piece. A further symbol of indefiniteness is provided, but it should remain highly improbable.

The following harmonic operator is referred to as *Harmonic Cadence*. Its goal is reducing the hard work of locating musical cadences to a series of simple logical formula. This operator assigns Boolean values to each possible chord sequence, in order to determine which chords may belong to a harmonic cadence. Thanks to TFR operator, now we can deal with every chord (that is a set of notes) as a single symbol in a very limited alphabet, so the task of Harmonic Cadence operator is simpler.

Finally, the *Harmonic Redundancy* operator provides an easy way to collapse redundant information.

Segmentation and MX Encoding: A Case Study

In order to show a practical result of our studies and implementations, a short example of MX encoding and segmentation is now provided.

The piece taken into consideration is a two-voices composition by J. S. Bach: Invention #4 BWV775. Due to the lack of space, only a short part of the whole piece can be shown.

In this example, we report the MX coding of musical contents, together with a number of other related representations. In particular, notational data can be retrieved from MX, whereas the graphical aspect of score is visible in TIFF format, its performance information is provided by MIDI layout, and the audio layer is represented by the corresponding waveform.

The example we give can be divided into two parts. In the former we want to underline the first key aspect of MX code: the "glue" function among different levels provided by the spine layer. This layer is situated inside `<logic>` tags. The latter part shows the effects of a possible segmentation and the corresponding MX representation. Within MX encoding, this part is marked by `<structural>` tags.

As regards the first section of the example, the one without segmentation, there we decided to include also the logical description of the first five score bars. In this way, the reader can have an idea of how score is encoded in MX, with a particular attention to chords and notes. Such information is embraced by `<los>` tags.

Some comments about pitch encoding, a new characteristic of MX 1.4. Pitch information is no more similar to NIFF one, that is relative to staff properties and related to the appearance in the printed score; nor we decided to move to an absolute but poor MIDI-like notation, considering only the frequency of notes and not their real notation inside the score. Our approach is more complex than the two previously listed, but also richer, and it provides a way to code both the absolute notation (the one which produces the physical sound) and the typographical aspect of notes. These aspects can be guessed through a careful analysis of `<notehead>` contents.

Besides, in order to present shortly the key features of our proposal, in the following example we highlighted also the references between our spine and a possible TIFF, MIDI and WAV file. These elements can be found inside `<notational>`, `<performance>` and `<audio>` tags.

The current MX version is 1.4, and the presented piece of XML file follows such standard.

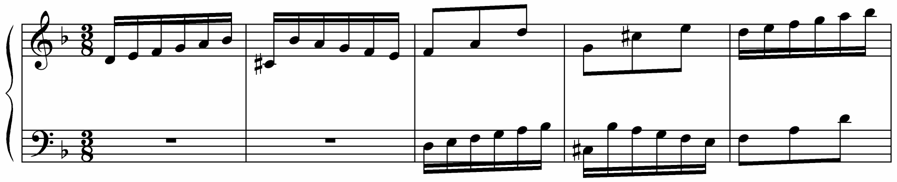


Fig. 3 – Notational layer: TIFF representation for Invention #4

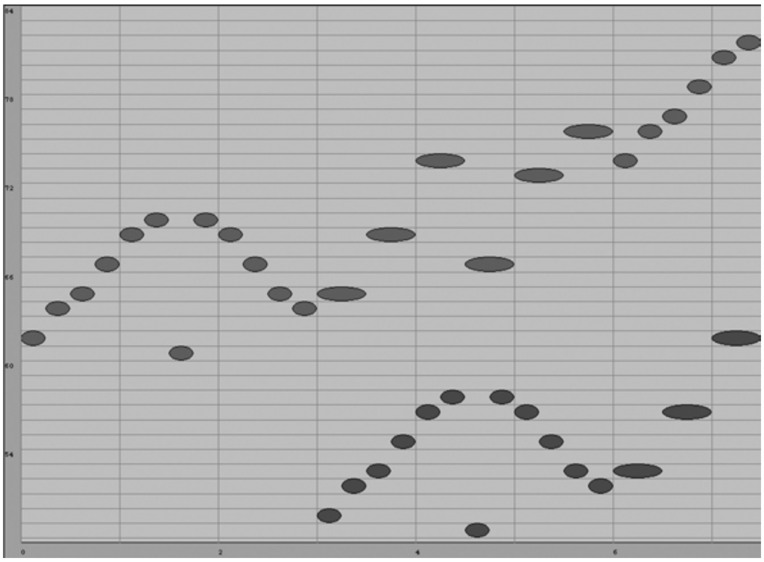


Fig. 4 – Performance layer: MIDI representation for Invention #4

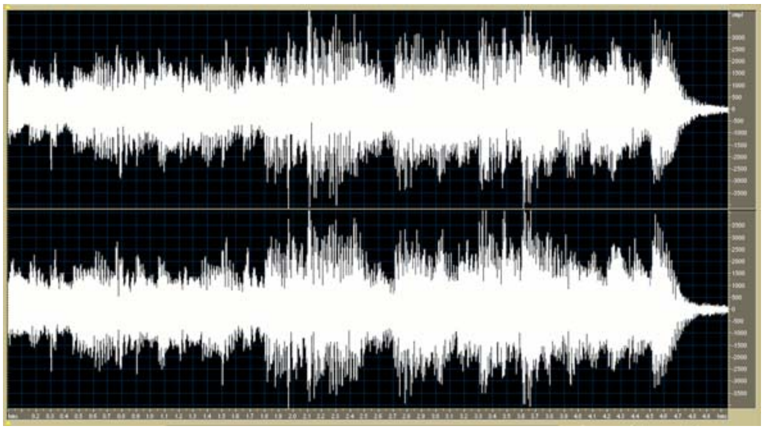


Fig. 5 – Audio layer: waveform representation for Invention #4

MX encoding of the first 5 bars of the piece, without segmentation:

```
<mx>
   <general>
      <description>
         <work_title>Zweistimmige Inventionen</work_title>
         <movement_title>Inventio #4</movement_title>
         <movement_number>BWV 774</movement_title>
         <genre>Baroque counterpoint</genre>
         <author>J.S. Bach</author>
      </description>
   </general>
```

```
<logic>
   <spine>
      <event id="e0" timing="0" hpos="0"/>
      <event id="v1_0" timing="0" hpos="110"/>
      <event id="v1_1" timing="256" hpos="300"/>
      <event id="v1_2" timing="256" hpos="300"/>
      <event id="v1_3" timing="256" hpos="300"/>
      <event id="v1_4" timing="256" hpos="300"/>
      <event id="v1_5" timing="256" hpos="300"/>
      <event id="v1_6" timing="256" hpos="450"/>
      <event id="v1_7" timing="256" hpos="300"/>
      <event id="v1_8" timing="256" hpos="300"/>
      <event id="v1_9" timing="256" hpos="300"/>
      <event id="v1_10" timing="256" hpos="300"/>
      <event id="v1_11" timing="256" hpos="300"/>
      <event id="v1_12" timing="256" hpos="400"/>
      <event id="v2_0" timing="0" hpos="0"/>
      <event id="v2_1" timing="256" hpos="300"/>
      <event id="v1_13" timing="256" hpos="300"/>
      <event id="v2_2" timing="0" hpos="0"/>
      <event id="v2_3" timing="256" hpos="300"/>
      <event id="v1_14" timing="256" hpos="350"/>
      <event id="v2_4" timing="0" hpos="0"/>
      <event id="v2_5" timing="256" hpos="300"/>
      <event id="v1_15" timing="256" hpos="600"/>
      <event id="v2_6" timing="0" hpos="0"/>
      <event id="v2_7" timing="256" hpos="300"/>
      <event id="v1_16" timing="256" hpos="350"/>
      <event id="v2_8" timing="0" hpos="0"/>
      <event id="v2_9" timing="256" hpos="300"/>
      <event id="v1_17" timing="256" hpos="350"/>
      <event id="v2_10" timing="0" hpos="0"/>
      <event id="v2_11" timing="256" hpos="300"/>
      <event id="v1_18" timing="256" hpos="500"/>
      <event id="v2_12" timing="0" hpos="0"/>
      <event id="v1_19" timing="256" hpos="200"/>
      <event id="v1_20" timing="256" hpos="200"/>
      <event id="v2_13" timing="0" hpos="0"/>
      <event id="v1_21" timing="256" hpos="200"/>
      <event id="v1_22" timing="256" hpos="200"/>
      <event id="v2_14" timing="0" hpos="0"/>
      <event id="v1_23" timing="256" hpos="200"/>
      …
   </spine>
<los>
   <staff_list>
      <staff id="staff0" ossia="no" line_number="5">
          <clef type="G" event_ref="e0" staff_step="2″
            octave_num="0"/>
      </staff>
      <staff id="staff1" ossia="no" linenumber="5">
          <clef type="F" event_ref="e0" staff_step="6"
            octave_num="0"/>
      </staff>
   </staff_list>
   <part id="part0" dfstaff_ref="staff0">
      <voice_list>
         <voice_item id="voice0"/>
      </voice_list>
      <measure number="1">
         <voice ref="voice0" ossia="no">
```

```
            <chord event_ref="v1_0" cue="no" grace="no">
               <notehead staff_ref="staff0">
                  <pitch step="D" octave="5"/>
                  <duration den="16" num="1"/>
               </notehead>
             </chord>
            <chord event_ref="v1_1" cue="no" grace="no">
               <notehead staff_ref="staff0">
                  <pitch step="E" octave="5"/>
                  <duration den="16" num="1"/>
               </notehead>
            </chord>
            <chord event_ref="v1_2" cue="no" grace="no">
               <notehead staff_ref="staff0">
                  <pitch step="F" octave="5"/>
                  <duration den="16" num="1"/>
               </notehead>
            </chord>
            …
         </voice>
         …
      </measure>
      …
   </part>
   <part id="part1" dfstaff_ref="staff1">
      <voice_list>
         <voice_item id="voice1"/>
      </voice_list>
      <measure number="3">
         <voice ref="voice1" ossia="no">
            <chord event_ref="v2_0" cue="no" grace="no">
               <notehead staff_ref="staff1">
                  <pitch step="D" octave="4"/>
                  <duration den="16" num="1"/>
               </notehead>
             </chord>
            <chord event_ref="v2_1" cue="no" grace="no">
               <notehead staff_ref="staff0">
                  <pitch step="E" octave="4"/>
                  <duration den="16" num="1"/>
               </notehead>
            </chord>
            <chord event_ref="v2_2" cue="no" grace="no">
               <notehead staff_ref="staff1">
                  <pitch step="F" octave="5"/>
                  <duration den="16" num="1"/>
               </notehead>
            </chord>
            …
         </voice>
         …
      </measure>
      …
   </part>

   …
</los>
<notational>
   <graphic_instance file_name="inventio4.tif" format="TIFF"
         spine_start_ref="v1_0" spine_end_ref="v1_23">
      <part_ref part_id=""/>
```

```
        <rights/>
      </graphic_instance>
    </notational>
    <performance>
      <performance_instance file_name="inventio4.mid"
            spine_start_ref="v1_0" spine_end_ref="v1_23">
        <MIDI format="0">
          <MIDI_part_ref part_id="Piano" track="1" channel="1"/>
        </MIDI>
        <rights/>
      </performance_instance>
    </performance>
    <audio>
      <clip file_name="inventio4.wav" format="PCM" duration="65.127"
            encoding="WAV" freq="44100" nbit="16"
            n_channel="2" spine_start_ref="v1_0" spine_end_ref="v1_23">
        <part_ref part_id=""/>
        <rights/>
        <index time="0.00" measure="1" beat="1" event_ref="v1_0"/>
        <index time="0.15" measure="1" beat="1" event_ref="v1_1"/>
        <index time="0.30" measure="1" beat="2" event_ref="v1_2"/>
        <index time="0.45" measure="1" beat="2" event_ref="v1_3"/>
        <index time="0.60" measure="1" beat="3" event_ref="v1_4"/>
        <index time="0.75" measure="1" beat="3" event_ref="v1_5"/>
        <index time="0.90" measure="2" beat="1" event_ref="v1_6"/>
        <index time="1.05" measure="2" beat="1" event_ref="v1_7"/>
        <index time="1.20" measure="2" beat="2" event_ref="v1_8"/>
        <index time="1.35" measure="2" beat="2" event_ref="v1_9"/>
        <index time="1.50" measure="2" beat="3" event_ref="v1_10"/>
        <index time="1.65" measure="2" beat="3" event_ref="v1_11"/>
        <index time="1.80" measure="3" beat="1" event_ref="v1_12"/>
        <index time="1.80" measure="3" beat="1" event_ref="v2_0"/>
        <index time="1.95" measure="3" beat="1" event_ref="v2_1"/>
        <index time="2.10" measure="3" beat="2" event_ref="v1_13"/>
        <index time="2.10" measure="3" beat="2" event_ref="v2_2"/>
        <index time="2.25" measure="3" beat="2" event_ref="v2_3"/>
        <index time="2.40" measure="3" beat="3" event_ref="v1_14"/>
        <index time="2.40" measure="3" beat="3" event_ref="v2_4"/>
        <index time="2.55" measure="3" beat="3" event_ref="v2_5"/>
        <index time="2.70" measure="4" beat="1" event_ref="v1_15"/>
        <index time="2.70" measure="4" beat="1" event_ref="v2_6"/>
        <index time="2.85" measure="4" beat="1" event_ref="v2_7"/>
        <index time="3.00" measure="4" beat="2" event_ref="v1_16"/>
        <index time="3.00" measure="4" beat="2" event_ref="v2_8"/>
        <index time="3.15" measure="4" beat="2" event_ref="v2_9"/>
        <index time="3.30" measure="4" beat="3" event_ref="v1_17"/>
        <index time="3.30" measure="4" beat="3" event_ref="v2_10"/>
        <index time="3.45" measure="4" beat="3" event_ref="v2_11"/>
        <index time="3.60" measure="5" beat="1" event_ref="v1_18"/>
        <index time="3.60" measure="5" beat="1" event_ref="v2_12"/>
        <index time="3.75" measure="5" beat="1" event_ref="v1_19"/>
        <index time="3.90" measure="5" beat="2" event_ref="v1_20"/>
        <index time="3.90" measure="5" beat="2" event_ref="v2_13"/>
        <index time="4.05" measure="5" beat="2" event_ref="v1_21"/>
        <index time="4.20" measure="5" beat="3" event_ref="v1_22"/>
        <index time="4.20" measure="5" beat="3" event_ref="v2_14"/>
        <index time="4.35" measure="5" beat="3" event_ref="v1_23"/>
        …
      </clip>
    </audio>
</mx>
```

The same score, or the same MX file, can be processed in order to find repetitive musical patterns and themes, according to the previously described operators. For instance, we can perform a segmentation based on the melodic operator of transposition. In this case, an automatic segmenter (with proper settings) or even a hand-made analysis process could recognize the two occurrences shaded in Fig. 6.

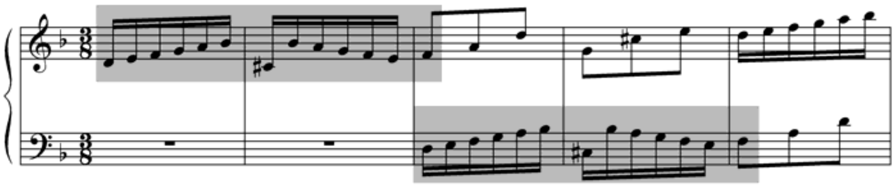


Fig. 6 – The shaded boxes contain the results of a possible segmentation

The new interesting part of the example is the one embraced by `<structural>` tags. Let's observe that the presence of a sub-element named `<analysis>` allows to encode in a single file multiple analyses, such as for instance those coming from different segmentation processes with different parameters and/or operators. Inside the single analysis, the format isolates a theme and lists all its occurrences, according to its original or variated form. As usual, the presence of references to the spine (in particular by `start_ref` and `end_ref` attributes) is fundamental: this is the way themes are mapped to the original score.

After taking segmentation into consideration, the resulting MX file would be modified as follows:

```
<mx>
   <general> … </general>
   <los> … </los>
   <structural>
      <analysis>
         <theme id="theme0">
            <occurrence>
               <thm_spine_ref end_ref="v1_12" part_ref="part0"
                  start_ref="v1_0" voice_ref="voice0"/>
            </occurrence>
            <occurrence>
               <thm_spine_ref end_ref="v2_12" part_ref="part1"
                  start_ref="v2_0" voice_ref="voice1"/>
            </occurrence>
         </theme>
         …
      </analysis>
   </structural>
   …
</mx>
```

Conclusion

In this paper we discussed two different aspects of music segmentation:

1. the capability to perform an automatic analysis process, based on flexible operators, appropriate data structures, and efficient algorithms;
2. the aptitude to represent the results of (either manual or automatic) segmentation in a comprehensive file format, so that those results can be linked not only to notational level but also to audio, performance, and graphic layers.

Our latest studies in the apparently independent fields of music XML and automatic segmentation originated a very interesting common outcome: the representation of music data and meta-data at any level of abstraction through MX, that revealed to be a comprehensive and effective standard format.

Acknowledgements

The authors wish to acknowledge the partial support of this project by Italian MIUR (FIRB "Web-Minds" project #RBNE01WEJT_005) and the Italian National Research Council, in the framework of the research program "Methodologies, techniques, and computer tools for the preservation, the structural organization, and the intelligent query of musical audio archives stored on heterogeneous magnetic media", Finalized Project "Cultural Heritage", Subproject 3, Topic 3.2, Subtopic 3.2.2, Target 3.2.1. We also want to acknowledge the members of the IEEE MX WG (PAR1599) for their cooperation and interest in our work.

This paper has been made possible by the efforts of researchers and graduate students of LIM.

References

1. *Definition of a Commonly Acceptable Musical Application Using the XML Language*, http://www.lim.dico.unimi.it/ieee/xml.html
2. *LIM Official WebSite*, http://www.lim.dico.unimi.it
3. Bertino, E., Catania B., and Ferrari, E., *Multimedia IR: Models and Languages. Modern Information Retrieval*, Addison-Wesley, 1999
4. Brinkman, A.R., *PASCAL Programming for Music Research*, The University of Chicago Press, Chicago, 1990
5. D'Onofrio, A., *Methods for Integrated Timed Audio and Textual Digital Music Processing*, Master Thesis, Computer Science Department, University of Milan, 1999
6. Frazzini, G., Haus, G., and Pollastri, E., *Cross Automatic Indexing of Score and Audio Sources: Approaches for Music Archive Applications*, Proceedings of the ACM SIGIR '99 Music Information Retrieval Workshop, Berkeley, 1999
7. Guagnini, S., *Strategies and Tools for the Automatic Segmentation of Music*, Master Thesis, Computer Science Department, University of Milan, 1998
8. Haus, G., and Longari, M., *Towards a Symbolic/Time-Based Music language based on XML*, IEEE Proceedings of MAX2002, Milan, 2002
9. Haus, G., and Pollastri, E., *A Multimodal Framework for Music Inputs*, in Proceedings of the ACM Multimedia, ACM Press, Los Angeles, 2000

10. Lerdahl, F., and Jackendoff, Ray, *A Generative Theory of Tonal Music*, The MIT Press, Cambridge, 1983
11. Longari, M., *Formal and software tools for a commonly acceptable musical application using the XML language*, Ph. D. Thesis, Computer Science Department, University of Milan, 2003
12. Ludovico, L.A., *Automatic Identification of Musical Structures: Models and Software Prototypes*, Master Thesis, Informatics Engineering Department, Politecnico of Milan, 2003
13. Polansky, L., *Morphological Metrics*, Journal of New Music Research, vol. 25, pages 289-368, Swets & Zeilinger Publ., Amsterdam, 1996

Evolutionary Optimization of Music Performance Annotation

Maarten Grachten, Josep Lluís Arcos, and Ramon López de Mántaras

IIIA, Artificial Intelligence Research Institute,
CSIC, Spanish Council for Scientific Research,
Campus UAB, 08193 Bellaterra, Catalonia, Spain,
{maarten,arcos,mantaras}@iiia.csic.es,
http://www.iiia.csic.es

Abstract. In this paper we present an enhancement of edit distance based music performance annotation. The annotation captures musical expressivity not only in terms of timing deviations but also represents e.g. spontaneous note ornamentation. To reduce the number of errors in automatic performance annotation, some optimization is essential. We have taken an evolutionary approach to optimize the parameter values of cost functions of the edit distance. Automatic optimization is desirable since manual parameter tuning is unfeasible when more than a few performances are taken into account. The validity of the optimized parameter settings is shown by assessing their error-percentage on a test set.

1 Introduction

Although the use of the edit distance [7] is well known in the field of melodic similarity [8,12], score following/automatic accompaniment [3,10] and performance transcription [6,9], not much attention has been paid to its value for the expressive analysis and annotation of musical performances. The optimal alignment between score and performance does not only reveal timing deviations of performed notes, but (depending on the set of edit operations) conveys a much richer set of expressive variations, such as ornamentations, and fragmentations/consolidations. In the context of the *ProMusic* project[1] we are developing *Tempo Express*, a Case Based Reasoning system for applying tempo transformations to audio recordings of solo performances of jazz melodies [5]. In this system, we use the alignment information to automatically annotate performances [1]. The performance annotations serve as example cases to transform a performance for a given melody. As a result, the expressiveness of the transformed performance is not restricted to timing variations, but it can also contain for example ornamentations.

For a correct detection of phenomena such as ornamentations, fragmentations, and consolidations of notes using the edit distance, it is important to

[1] MCyT. TIC2003-07776-C2-02

assign appropriate costs to each of the edit operations. Since it turned out to be unfeasible to manually tune the costs to obtain correct annotations for a large set of performances, we tried to find good costs using a genetic algorithm. In this paper, we describe our experiments and results.

In section 2, we explain the idea of annotating performances by *performance events*. We wil show how performance annotations can be constructured using the edit distance algorithm, and motivate the chosen set of cost functions for assessing the costs of the edit operations. In section 3, we report how the parameter values in the cost functions were estimated, using a genetic algorithm, and evaluate the quality of the estimations. Conclusions are presented in section 4.

2 Performance Annotation

It has been widely acknowledged that human performances of musical material are virtually always quite different from mechanical renderings of the music. These differences (the musical expressivity) are thought to be vital for the aesthetic quality of the performance, and therefore it is worthwhile to have ways of making explicit the quality and quantity of these differences. The majority of research concerning musical expressivity is focused on the temporal, or dynamic variations of the notes of the musical score as they are performed [2,4,11,13]. In this context, the spontaneous insertions or deletions of notes by the performer are often discarded as artifacts, or performance errors. This may be due to the fact that most of this research is focused on the performance practice of classical music, where the interpretation of notated music is rather strict. Contrastingly, in jazz music performers often favor a more liberal interpretation of the score, so that expressive variation is not limited to variations in timing of score notes, but also comes in the form of e.g. deliberately inserted and deleted notes. Thus, research concerning expressivity in jazz music should pay heed to these phenomena and in addition to capturing the temporal/dynamical variations of score notes, the musical behavior of the performer should be described in terms of note insertions/deletion/ornamentations etcetera. One way to do this is to define these expressive phenomena as *performance events*, and then annotate the performance with a sequence of such events.

In the next subsections, we propose a set of performance events to be used in the performance annotation of saxophone jazz performances, we show how the edit distance algorithm can be used to construct the annotation, and propose cost functions to be used in the edit distance computation.

2.1 Choice of Performance Events

The decision which performance events to define is important, since they are in a sense the 'vocabulary' we use to represent the musician's performance behavior. The set of performance events proposed here (which is a slight extension of the one proposed in Arcos et. al. [1]), is chosen to reflect the variety of phenomena that we have actually encountered in a set of saxophone jazz performances.

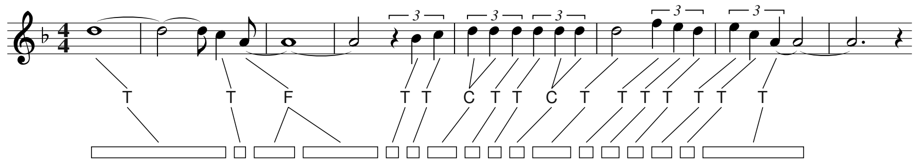


Fig. 1. performance annotation of the first phrase of *Once I Loved*, played by a saxophone at 220 bpm. The bars below denote the played notes (the bar lengths are representative for the note durations). The annotation is the sequence of performance events in the middle. 'T' is for transformation (of duration and onset), 'F' for fragmentation, and 'C' for consolidation

Based on the fact that the phrases in our data set were played by a professional musician and they were performed with the intention of giving a neutral interpretation of the score, it may be thought that expressive deviations of the score other than changing the timing or dynamics of the notes will occur only very infrequently, and that most of the performances can be represented by events that just describe how the timing and duration of score notes was changed as they were performed – i.e. *transformation* events. But listening to the performances revealed that other types of events occurred frequently as well. For example, some cases of note deletions and insertions were found. Apart from real insertions of notes, that gave the impression of an elaboration of the melody (such insertions occurred, but were rare), another type of insertion was found to occur rather often: ornamentation. By ornamentation we refer to one or more very short notes (typically about 100 or 200 ms.) that are usually a chromatic approach from below to the next score note. We have found such ornamentations to consist of one, two or three notes. Furthermore, we observed that consolidation (as described in the previous section) occurred in some performances. Occasionally, we found cases of fragmentation. Other transformations, such as sequential transposition (reversal of the temporal order of notes) were not encountered.

For illustration, figure 1 shows the annotation of a melodic phrase from the song 'Once I Loved' (A.C. Jobim). The performer fragmented the third note into two shorter notes, and in the repetition of triplet notes, there are two cases of consolidations. The other notes were played as is, with a lesser or greater degree of deviation in onset time and duration.

The kinds of events mentioned above can be visualized in a class hierarchy (as in figure 2) to make explicit their characteristics. The core idea of a performance event is that it relates the notes that are actually played by the performer to the notes that are written in the score. As such, every event refers either to one or more notes in the score, or to one or more notes in the performance, or both. We can distinguish, within this general class of *reference* events, those that refer to notes in the score and those that refer to elements in the performance. *Deletion* events refer to notes of the score that are not present in the performance (i.e. the notes that are not played), therefore they can be classified

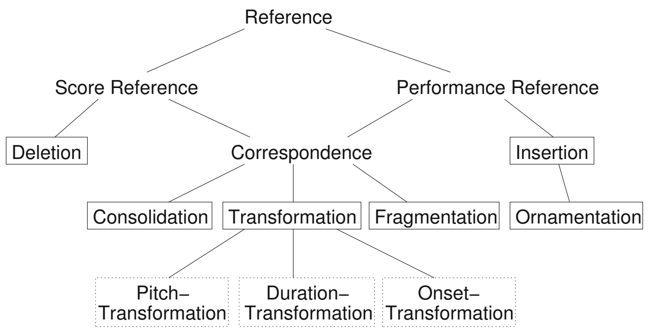


Fig. 2. A hierarchical representation of performance events for performance annotation. The unboxed names denote abstract classes; the boxed names denote 'concrete' classes that are used in the performance annotation. The dottedly boxed names denote classes that are derived from the concrete classes

as *score-reference* events. Conversely, *insertion* events refer only to elements in the performance (i.e. the notes that were added), so they form a subclass of *performance-reference* events. *Transformation*, *consolidation* and *fragmentation* events refer to elements from both the score and the performance and thus form a shared subclass of *score-reference* and *performance-reference* events. We call this class *correspondence* events.

The *reference*, *score-reference*, *performance-reference* and *correspondence* classes are abstract classes that are just conceived to express the relationships between concrete classes of events, and are not intended to be used directly in the performance annotation. The concrete classes, that are used to construct the performance annotation, are depicted in figure 2 with boxes. They are:

Insertion Represents the occurrence of a performed note that is not in the score
Deletion Represents the non-occurrence of a score note in the performance
Consolidation Represents the agglomeration of multiple score notes into a single performed note
Fragmentation Represents the performance of a single score note as multiple notes
Transformation Represents the change of nominal note features
Ornamentation Represents the insertion of one or several short notes to anticipate another performed note

In the case of *transformation*, we are not only interested in the one-to-one correspondence of performance elements to score elements itself, but rather in the changes that are made to attribute values of score notes when they are transformed into performance elements. Therefore, we view transformation events as compositions of several transformations, e.g. *pitch transformations*, *duration transformations* and *onset transformations*.

2.2 Constructing Performance Annotations Using Edit Distance

Performance annotations, being sequences of performance events, can be treated as a sequence of operations that tell you how to perform a written score, or alternatively, how to transform a sequence of score notes into a sequence of performed notes. As such, the relation between the performance annotation and the concept of *optimal alignment* between score and performance becomes obvious. When performance events are defined as edit operations, then computing the edit distance between the score and the performance yields a sequence of edit operations, from which the performance annotation is constructed.

One problematic aspect of defining the performance events as edit operations, is the fact that the subclasses of *transformation* events (*pitch transformations*, *duration transformations* and *onset transformations*), can occur simultaneously (that is, they can refer to the same score and performance notes), whereas in the edit distance, each sequence element is covered by exactly one operation. Our solution is to have a single Transformation operation for the computation of the alignment, as a rough identification of the expressivity. In a second stage, after the alignment has been computed, the score and performance events corresponding to Transformation operations can be compared in more detail to establish which of the pitch, duration, and onset transformation really occurred. The corresponding classes are shown in figure 2 as dotted boxes.

The edit distance between a source and a target sequence is defined as the minimum cost of transforming the source sequence into the target sequence using a fixed set of edit operations. This cost can be calculated using the following recurrence equation, that defines the distance $d_{m,n}$ between two sequences $\langle a_1, a_2, ..., a_m \rangle$ and $\langle b_1, b_2, ..., b_n \rangle$ (using insertion, deletion and replacement, the standard set of edit operations):

$$d_{i,j} = min \begin{cases} d_{i-1,j} + w(a_i, \emptyset) & \text{(deletion)} \\ d_{i,j-1} + w(\emptyset, b_j) & \text{(insertion)} \\ d_{i-1,j-1} + w(a_i, b_j) & \text{(replacement)} \end{cases}$$

for all $0 \leq i \leq m$ and $0 \leq j \leq n$, where m is the length of the source sequence and n is the length of the target sequence. The initial conditions for the recurrence equation are:

$$\begin{aligned} d_{i,0} &= d_{i-1,j} + w(a_i, \emptyset) & \text{(deletion)} \\ d_{0,j} &= d_{i,j-1} + w(\emptyset, b_j) & \text{(insertion)} \\ d_{0,0} &= 0 \end{aligned}$$

The weight function w, defines the cost of operations, such that e.g. $w(a_4, \emptyset)$ returns the cost of deleting element a_4 from the source sequence, and $w(a_3, b_5)$ returns the cost of replacing element a_3 from the source sequence by the element b_5 of the target sequence.

For two sequences a and b, consisting of m and n elements respectively, the values $d_{i,j}$ (with $0 \leq i \leq m$ and $0 \leq j \leq n$) are stored in an $n + 1$ by $m + 1$ matrix. The value in the cell at the lower-right corner, $d_{m,n}$ is taken as the distance between a and b, that is, the minimal cost of transforming the sequence $\langle a_0, ..., a_m \rangle$ into $\langle b_0, ..., b_n \rangle$.

In our particular case, we define the following edit operations: insertion, deletion, ornamentation, transformation, fragmentation, and consolidation. To use these operations for score performance alignment, cost values must be assigned to each of them. This is done by means of weight functions for each type of operation (w in the recurrence equation).

2.3 The Cost Values

Ideally, the cost values for each type of operation will be such that the resulting optimal alignment corresponds to an intuitive judgment of how the performance aligns to the score (in practice, the subjectivity and ambiguity that is involved in establishing this mapping by ear, turns out to be largely unproblematic). The main factors that determine which of all the possible alignments between score and performance is optimal, will be on the one hand the features of the note elements that are involved in calculating the cost of applying an operation, and on the other hand the relative costs of the operations with respect to each other.

In establishing which features of the compared note elements are considered in the comparison, we have taken the choices made by Mongeau and Sankoff [8] as a starting point. In addition to pitch and duration information (proposed by Mongeau and Sankoff), we have decided to incorporate the difference in position in the costs of the correspondence operations (transformation, consolidation and fragmentation), because this turned out to improve the alignment in some cases. One such case occurs when one note in a row of notes with the same pitch and duration is omitted in the performance. Without taking into account positions, the optimal alignment will delete an arbitrary note of the sequence, since the deletions of each of these notes are equivalent based on pitch and duration information only. When position *is* taken into account, the remaining notes of the performance will all be mapped to the closest notes in the score, so the deletion operation will be performed on the score note that remains unmapped, which is often the desired result.

It is important to note that when combining different features, like pitch, duration and onset into a cost-value for an operation, the relative contribution of each term is rather arbitrary. For example when the cost of transforming one note into another would be defined as the difference in pitch plus the difference in duration, the outcome depends on the units of measure for each feature. The relative weight of duration and pitch is not the same when measured in seconds, as when measured in beats. Similarly, pitch could be measured in frequency, semitones, scale steps, etcetera. Therefore, we have chosen a parametrized approach, in which the relative contribution of each term in the weight function is weighted by a constant parameter value.

The other aspect of designing cost-functions is the relative cost of each operation. After establishing the formula for calculating the weights of each operation, it may be that some operations should be systematically preferred to others. This independence of costs can be achieved by multiplying the cost of each operation by a factor and adding a constant.

The cost functions w for the edit operations are given below. The arguments of the functions are elements from a sequence of score notes s, and a sequence of performed notes p. $\mathcal{P}$, $\mathcal{D}$, and $\mathcal{O}$ are functions such that $\mathcal{P}(x)$ returns the pitch (as a MIDI number) of a score note or performed note x, $\mathcal{D}(x)$ returns its duration, and $\mathcal{O}(x)$ returns its onset time. Equations 1, 2, 3, 4, 5, 6 define the costs of deletion, insertion, ornamentation, transformation, consolidation and fragmentation, respectively.

$$w(s_i, \emptyset) = \alpha_d \cdot \mathcal{D}(s_i) \tag{1}$$

$$w(\emptyset, p_j) = \alpha_i \cdot \mathcal{D}(p_j) \tag{2}$$

$$w(\emptyset, p_j, \cdots, p_{j+L+1}) = \alpha_o \cdot \begin{pmatrix} \beta \cdot \sum_{l=1}^{L} |1 + \mathcal{P}(p_{j+l}) - \mathcal{P}(p_{j+l-1})| + \\ \gamma \cdot \sum_{l=0}^{L} \mathcal{D}(p_{j+l})| \end{pmatrix} \tag{3}$$

$$w(s_i, p_j) = \alpha_t \cdot \begin{pmatrix} \beta \cdot |\mathcal{P}(s_i) - \mathcal{P}(p_j)| + \\ \gamma \cdot |\mathcal{D}(s_i) - \mathcal{D}(p_j)| + \\ \delta \cdot |\mathcal{O}(s_i) - \mathcal{O}(p_j)| \end{pmatrix} \tag{4}$$

$$w(s_i, ..., s_{i+K}, p_j) = \alpha_c \cdot \begin{pmatrix} \beta \cdot \sum_{k=0}^{K} |\mathcal{P}(s_{i+k}) - \mathcal{P}(p_j)| + \\ \gamma \cdot |\mathcal{D}(p_j) - \sum_{k=0}^{K} \mathcal{D}(s_{i+k})| + \\ \delta \cdot |\mathcal{O}(s_i) - \mathcal{O}(p_j)| \end{pmatrix} \tag{5}$$

$$w(s_i, p_j, ..., p_{j+L}) = \alpha_f \cdot \begin{pmatrix} \beta \cdot \sum_{l=0}^{L} |\mathcal{P}(s_i) - \mathcal{P}(p_{j+l})| + \\ \gamma \cdot |\mathcal{D}(s_i) - \sum_{l=0}^{L} \mathcal{D}(p_{j+l})| + \\ \delta \cdot |\mathcal{O}(s_i) - \mathcal{O}(p_j)| \end{pmatrix} \tag{6}$$

The parameters β,γ, and δ control the influence of pitch, duration, and onset, respectively. α_d, α_i, α_o, α_t, α_c, and α_f are the parameters that scale the costs of deletion, insertion, ornamentation, transformation, consolidation and fragmentation, respectively.

The costs of transformation (4), consolidation (5), and fragmentation (6), are principally constituted of the differences in pitch, duration and onset times between the compared elements. In the case of one-to-many matching (fragmentation) or many-to-one (consolidation), the difference in pitch is calculated as the sum of the differences between the pitch of the single element and the pitches of the multiple elements. The difference in duration is computed between the duration of the single element and the sum of the durations of the multiple elements. The difference in onset is computed between the onset of the single element and the onset of the onset of the first of the multiple elements. The cost of deletion (1) and insertion (2)is determined by the duration of the deleted element. The cost of ornamentation (3) is determined by the pitch relation of the ornamentation elements and the ornamented element (chromatically ascending sequences are preferred), and the total duration of the ornamentation elements.

3 Experimentation

The introduction of the nine parameters in the cost functions comes with the problem of finding appropriate values for those parameters. Although the edit distance has some robustness (it aligns sequences reasonably well, even if bad parameter values are chosen), it is difficult to bring the amount of annotation errors down to a few percent. Manually tuning the parameters is possible for a small set of performances, but this becomes unfeasible for larger sets (adjustments that improve the annotation of one performance, worsened the annotation of others). Surprisingly, our manually tuned settings hardly improved the accuracy of annotation with respect to random parameter settings when tested on larger sets of performances. Therefore, we have employed a genetic algorithm to obtain a good parameter setting. In this section we describe our experimentation with the tuning of the parameters.

The idea of the evolutionary optimization of the parameter values is rather simple: an array of the nine parameter values (one value for each parameter) can be treated as a chromosome. The number of errors produced in the annotation of a set of performances using that set of parameter values, is inversely related to the fitness of the chromosome. By evolving an initial population of (random) chromosomes through crossover, mutation and selection, we expect to find a set of parameter values that minimizes the number of annotation errors, and thus improves automatic performance annotation.

We are interested in two main questions. The first is whether it is possible to find a parameter setting that works well in general. That is, can we expect a parameter setting that worked well for a training set to perform well on unseen performances? The second question is whether there is a single setting of parameter values that optimizes the annotations. It is also conceivable that good annotations can be achieved by several different parameter settings.

3.1 Experiment Setup

We have run the genetic algorithm with two different (non-overlapping) training sets, both containing twenty performances. These were (monophonic) saxophone performances of eight different phrases from two jazz songs (*Body and Soul*, and *Once I Loved*), performed at different tempos. For each of the performances, the correct annotation was available. The fitness of the populations was assessed using these annotations.

The fitness evaluation of a population (consisting of 20 chromosomes) on the training set is a rather time consuming operation. Therefore, it can take a long time before a good solution is obtained, starting the evolution with a randomly initialized population. In an attempt to solve this problem, we initialized the population with solutions that were trained on the individual phrases of the training set (which is a much faster procedure). Assuming that the solution optimized for one phrase may in some cases work for other phrases, this speeds up the time needed to find a good solution for the whole training set.

A new generation is generated from an old generation as follows: From the old generation (consisting of N chromosomes), the k best chromosomes are selected (where k is dependent on the distribution of the fitness across the population); Then, $N - k$ new chromosomes are created by a cross-over of the selected chromosomes; The newly generated chromosomes are mutated (multiplying each parameter value by a random value), and the $N - k$ mutated chromosomes, together with the n (unchanged) chromosomes from the old generation, form the new generation.

3.2 Fitness Calculation

The fitness of the chromosomes is calculated by counting the number of annotation errors using the parameter values in the chromosome. For example, assume that the correct annotation of a melodic fragment is 'T T C T', and the annotation of that fragment obtained by using the parameter values of the chromosome is 'T T T D T' (that is, a consolidation operation is confused with an transformation and a deletion operation). The 'C' doesn't match to an element in the second sequence, and the 'T' and 'D' don't match to elements in the first sequence and thus three errors occur. To count the errors between the correct and the predicted annotations (which are represented as sequences of symbols), we use the edit distance (don't confuse this use of the edit distance to *compare* annotations with the use of the edit distance to *generate* annotations).

For a given set S of performances (for which the correct annotations are known), we define the fitness of a chromosome c as:

$$\mathit{fit}(c) = \frac{1}{E(c,S)+1}$$

where $E(c, S)$ is the total number of errors in the predicted annotations for S using the parameter values in c. The fitness function *fit* ranges from zero to one. Obviously, a fitness value of one is the most desirable, since it corresponds to zero annotation errors.

3.3 Results

For each of the two training sets, Tr1 and Tr2, the evolution algorithm was run three times. The resulting parameter settings are shown in figure 3. Table 1 shows the number of annotation errors each of the parameter settings produced on the training sets, and on a test set (a set of 35 performances, none of which occurred in Tr1 or Tr2). The average number of annotation errors on the test set is about 32 on a total of 875 annotation elements in the test set, an error-percentage of 3.66%. This is only slightly higher than the error-percentages on

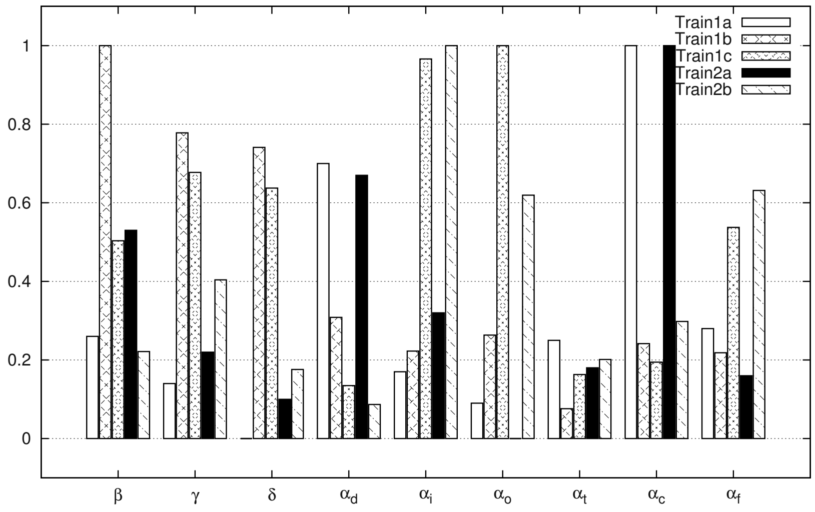


Fig. 3. Estimated parameter values for two different training sets (Tr1 and Tr2). Three runs were done for each set (a, b, and c). The x-axis shows the nine different parameters of the cost functions (see section 2.3). For each parameter the values are shown for each run on both training sets

the training sets: 2,60% for Tr1, and 2,37% for Tr2 (averaged over three runs), and substantially lower than the average error-percentage of random parameter settings on the test set, which is about 13,70%.

	Tr1a	Tr1b	Tr1c	Tr2a	Tr2b	Tr2c
Errors on Train	19 (3.89)	9 (1.84)	10 (2.05)	11 (2.30)	12 (2.51)	11 (2.30)
Errors on Test	19 (2.17)	26 (2.97)	30 (3.43)	19 (2.17)	32 (3.66)	65 (7.43)

Table 1. Annotation errors produced by the obtained solutions for three different runs (denoted by the letters a, b, and c) on two different training sets (Tr1 and Tr2) and a test set. The first row shows the number of errors on the set that the solutions were trained on, and the corresponding percentages in parentheses (Tr1 contained 488 annotation elements in total, and Tr2 contained 479). The second row shows the number of errors on the test set (875 elements), with percentages in parentheses

Table 2 shows the pair-wise correlations of the values. As can be seen from the cross-correlations in the table, the parameter settings did not all converge to the same values. Nevertheless, there were some cases in which the parameters were highly correlated. In particular the solutions found in runs Tr1a, and Tr2a are highly similar (this can be easily verified by eye in figure 3. A rather strong correlation is also observed between the solutions found in Tr1c and Tr2b, and those in Tr1b, and Tr2c. It is interesting that the correlated solutions were obtained using non-overlapping sets of performances. This is evidence that the solutions found are approximations of a single parameter setting that is valid for the performances in both training sets. In the case of the solutions of Tr1a and Tr2a, the approximated parameter setting may also have a more general validity, since both solutions have a low error number of annotations on the test set as well (see table 1).

	Tr1a	Tr1b	Tr1c	Tr2a	Tr2b	Tr2c
Tr1a	1.00	-0.32	-0.70	0.92	-0.32	-0.28
Tr1b	-0.32	1.00	0.17	-0.02	-0.33	0.68
Tr1c	-0.70	0.17	1.00	-0.61	0.76	0.07
Tr2a	0.92	-0.02	-0.61	1.00	-0.33	-0.12
Tr2b	-0.32	-0.33	0.76	-0.33	1.00	-0.47
Tr2c	-0.28	0.68	0.07	-0.12	-0.47	1.00

Table 2. Cross-correlations of the parameter values that were optimized using two different training sets (Tr1 and Tr2), and three runs for each set (a, b, and c)

4 Conclusions and Future Work

We have presented a method to enhance the automatic annotation of music performances. The annotation includes information about e.g. note ornamentations and deletions as part of the musical expressivity. To correctly detect such phenomena, an evolutionary approach was chosen to optimize the parameter values of cost functions, that were used in the (edit distance based) performance annotation process.

Two main questions we have tried to answer is whether it is possible to find a parameter setting that has a broader validity than just the set of performances it was optimized for, and whether there is a single parameter setting that optimizes the annotations. All solutions from different trials on two non-overlapping sets of performances substantially improved the quality of annotation of a test set over random parameter settings. Moreover, cross-correlations were found between some parameter settings that were optimized for different training sets. This suggests that they are approximations of a parameter setting that works well for a larger group of performances. In general however, the solutions did not all converge to a single set of parameter values.

In the future, we wish to extend the experiments to see whether the solutions found converge to a limited range of parameter settings. And if so, we wish to investigate how the distributions of values over the parameters relate to each other (for example, do high α_c values imply low γ and δ values and vice versa?).

Acknowledgments

This research has been partially supported by the Spanish Ministry of Science and Technology under the project TIC 2003-07776-C2-02 "CBR-ProMusic: Content-based Music Processing using CBR" and EU-FEDER funds. The authors acknowledge the Music Technology Group of the Pompeu Fabra University for providing the analysis module that was used to extract note information from the audio files.

References

1. J. Ll. Arcos, M. Grachten, and R. López de Mántaras. Extracting performer's behaviors to annotate cases in a CBR system for musical tempo transformations. In *Proceedings of the Fifth International Conference on Case-Based Reasoning (ICCBR-03)*, 2003.
2. Sergio Canazza, Giovanni De Poli, Stefano Rinaldin, and Alvise Vidolin. Sonological analysis of clarinet expressivity. In Marc Leman, editor, *Music, Gestalt, and Computing: studies in cognitive and systematic musicology*, number 1317 in Lecture Notes in Artificial Intelligence, pages 431–440. Springer, 1997.
3. R. Dannenberg. An on-line algorithm for real-time accompaniment. In *Proceedings of the 1984 International Computer Music Conference*. International Computer Music Association, 1984.
4. P. Desain and H. Honing. Does expressive timing in music performance scale proportionally with tempo? *Psychological Research*, 56:285–292, 1994.
5. E. Gómez, M. Grachten, X. Amatriain, and J. Ll. Arcos. Melodic characterization of monophonic recordings for expressive tempo transformations. In *Proceedings of Stockholm Music Acoustics Conference 2003*, 2003.
6. E. W. Large. Dynamic programming for the analysis of serial behaviors. *Behavior Research Methods, Instruments & Computers*, 25(2):238–241, 1993.
7. V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10:707–710, 1966.
8. M. Mongeau and D. Sankoff. Comparison of musical sequences. *Computers and the Humanities*, 24:161–175, 1990.
9. B. Pardo and W. Birmingham. Improved score following for acoustic performances. In *International Computer Music Conference (ICMC)*, Goteborg, Sweden, 2002. The International Computer Music Association.
10. M. Puckette and A. C. Lippe. Score following in practice. In *Proceedings, International Computer Music Conference*, pages 182–185, San Francisco, 1992. International Computer Music Association.
11. B. H. Repp. Quantitative effects of global tempo on expressive timing in music performance: Some perceptual evidence. *Music Perception*, 13(1):39–58, 1995.
12. Ll. A. Smith, R. J. McNab, and I. H. Witten. Sequence-based melodic comparison: A dynamic programming approach. In W. B. Hewlett and E. Selfridge-Field, editors, *Melodic Similarity. Concepts, Procedures, and Applications*, Computing in Musicology, pages 101–118. MIT Press, 1998.
13. G. Widmer. Large-scale induction of expressive performance rules: First quantitative results. In *Proceedings of the International Computer Music Conference (ICMC2000)*, San Francisco, CA, 2000. International Computer Music Association.

Parichaykrama – An Exploratory Interface of Indian Classical Music Using Experiential Framework

Debopam Roy, Nagaraju Pappu, T.V. Prabhakar

Computer Science Department, IIT Kanpur,
Kanpur 208016, India
{debopam, npappu, tvp}@cse.iitk.ac.in

Abstract: - The design and development of large-scale repositories especially of content types like music pose challenging problems. We believe such databases should have exploratory interfaces centered on user experience as a navigation tool. Further the interface should not be constrained by the cognitive model of the developer and should be able to adapt to the user's own perception of how the content should be searched. We present such a system for Indian Classical Music. Here each musical piece is qualified by seven characteristics. The user can choose any one of them, at a time, to reduce the candidate collection. Musicons are used to help select the characteristic. Eliminating the characteristics one by one, the user arrives at the piece of his/her choice.

Keywords: - Exploratory interface, experiential interface, Indian classical music, musicon, musical smell.

1 Introduction

Handling large-scale repositories where there are more than a million entries is often a subject of great concern. Presenting a proper user interface for traversing such a humongous database takes a lot of thought. One is especially concerned with the aspect that more often than not, such databases give some optimum traversal routes while ignoring the special requirements for different categories of users. We propose that large collections, which mandate an interactive user interface, be explored via experiencing. We call it an experiential exploratory interface. Our model example is a repository of Indian classical music. In terms of complexity, Indian classical music is one of the most challenging subjects. If one uses tests like number of musical notes approved, basic musical scales accepted, musical instruments in circulation, rhythms used, methods governing the production of voice, current musical symbolism and a host of others, one will indeed see that the musical map overruns the political map of India. But then how do we qualitatively present a musical repository, which will cater to all tastes and needs. We need a suitable exploratory interface based on user experience which would map out all the major tenets of Indian classical music for the connoisseur as also help a lay man explore the repository. We also introduce an idea of musical "smell" which will lead the user to his destination point irrespective of the point from which he has started.

2 Information Handling in Large Databases

When we have large amounts of information, the organization of this information is always a key problem for information architects. Most often we tend to build structures that are static, and force the user to learn 'my cognitive structure' instead of a program or an interface that adapts to the users cognitive structure. This is ‘pro-builder’ approach and makes the user traverse the specific path that the creator has created for him. A more interactive interface which is ‘pro-user’ will give the choice of traversing a path to the user and still lead him to where he initially wanted to go. Now, the question is how and what we should do in order to build an information retrieval engine that can adapt to the users cognitive structure.

We propose a conceptual framework for facilitating the dynamic generation of the navigational model by the user at run time. We visualize the concept space to consist of a collection of attributes, which act as handles and help in uniquely identifying the piece. The other challenge is to generate this interface dynamically so that the user can decide on his/her route of traversal.

3 A “Sweet” Analogy

Before moving on to a description of how one may form an experiential framework for the exploration of Indian classical music, it will help if we give an example of what one wants to achieve from such a framework and how exactly it might work. Imagine yourself going to an Indian sweet shop. It is more than likely that you will be overwhelmed by the array of choices that are laid in front of you. You may be a connoisseur of Indian sweets in which case, you may ask for a special variety of a specific sweet made using a specific ingredient and in a specific quantity. But what if you are not? It is a well known strategy of the seller in such cases to let you taste a piece of his products. As you taste a piece, you make up your mind whether you would like to buy it. But how do you taste, say 50 varieties of sweets, laid in front of you? To ease your ‘navigation’ the seller may prompt you about whether you are allergic to honey (thus removing all sweets made using honey) or if you like ‘dry’ sweets (thus eliminating the ‘wet’ sweets like rasagolla) or ‘dry’ versions of ‘wet’ sweets, and then just to help you make up your opinion will allow you to taste a piece of both types. Once he has helped you to narrow your search, he may further ask questions like if you would prefer the ones with slight apricot spicing on top to further narrow down your choices and let you taste some of it again. The interface we are proposing is inspired from this and reflects this scenario with the user’s experience at each level of traversal helping in framing the ultimate route to his/her destination point. The dynamic generation of the questions is what forms the exploratory part of the interface.

4 Indian Classical Music

Indian classical music is graced by an astounding variety and richness of content. There are so many aspects to it that one will run out of space if one were to enumerate them all. For our purposes we have categorized each Indian classical piece to possess 7 basic characteristics [1]. These 7 characteristics are:

Type (vocal, instrumental or *jugalbandi*(mixture) of both),
Raga of the piece,
Tala of the piece,
Gharana of the piece
Name of the artist (who has performed it)
Region (hailing from which Indian region),
Version (whether a cover version or a live version etc).

The seven characteristics we have identified are described below with the further breakdown of each characteristic with a relational schema and description for each. It should be noted that the individual breakdown of each attribute is not an exclusive list. We have only pointed out the most well known as well the more interesting sub divisions for a characteristic.

Type: - A musical piece may be a pure vocal piece, to the accompaniment of suitable instruments or without any instruments, can be a pure instrumental piece (consisting of several or a single instrument) or even be a *jugalbandi* (a mixture of part vocal with part instrumental).
Its relational schema looks as follows.

Type {Vocal, Instrumental, Jugalbandi}
Vocal {Art Music, Light Music}
Art Music {Dhrupad, Khayal, Ras, Chaturanga, Tarana, Ashtapadi, Sargam Geet}
Light Music {Thumri, Dadra, Kajri, Chaiti, Sawan, Qawwali, Ghazal, Bhajan, Tappa}
Instrumental {String, Solid Bodied, Wind, Membrane Covered}
String {Dilruba, Sarangi, Israj, Violin}
Solid Bodied {Piano, Keyboard}
Wind {Flute, Shehnai, Clarinet, Harmonium}
Membrane Covered {Tabla, Pakhwaz, Dholak}

Raga – The raga is unique to Indian classical music and has no equivalent in western classical music. Roughly it may be described as a melodic framework which governs the way a piece is performed.
Its relational schema looks as follows.

Raga {Thaat, Aroha, Avaroha, Jati, Special Notes, Register, Pakad, Chalan, Position, Time of Singing}
Thaat {Bilawal, Kalyan, Khamaj, Bhairav, Poorvi, Marwa, Kafi, Asawari, Todi, Bhairavi}
Jati {Sampoorna, Shadava, Oudava, Sampoorna Shadava, Sampoorna Oudava, Shadava, Oudava, Shadava Sampoorna, Oudava Shadava, Oudava Sampoorna}
Register {Mandra, Madhya, Tara}

Position {Purvarang, Uttarang}
Time of Singing {Morning, Noon, Evening, Night, Anytime of day, Seasonal}

Tala – The *tala* refers to the fundamental concept of beat patterns to which a particular composition is set. Figuratively speaking, the *tala* assumes greater importance when it is pure instrumental piece. For vocal renditions, the raga is the more important constituent. However, every piece must be set to a unique *raga* and a unique *tala*.
Its relational schema looks as follows.
Tala {Madhya, Vilambit, Drut, Dugun, Pat, Theka, Tihai}

Gharana – Every musical piece has a history attached to it, which traces its beginning, its very conception. *Gharanas* can thus said to be the founding houses which gave certain characteristic features to a particular vocal or instrumental rendition. More generally speaking, *gharanas* framed the musicological ideology of the artistic piece.
Its relational schema looks as follows.
Gharana {vocal, instrumental}
Vocal {Khayal Gharanas, Thumri gharanas}
Khayal Gharanas {Gwalior, Agra, Jaipur, Kirana}
Thumri Gharanas {Benaras, Patiala}
Instrumental {Pakhwaj Gharanas, Tabla Gharanas, Sitar Gharanas}
Pakhwaj Gharanas {Kudau Singh, Nana Panse, Nathadwara}
Tabla Gharanas {Delhi, Ajrada, Lucknow, Farrukhabad, Benaras, Punjab}
Sitar Gharanas {Jaipur, Rampur/Mahiyar, Imdad Khan}

Artist – This refers to the artist who has performed the piece. In case there are multiple artists involved, a single person is identified as the main artist and the piece is listed under his name. In case of jugalbandis or duets, the piece can be accessed by either artist.
Its relational schema looks as follows.
Artist {Male, Female, Bands}
Male {by name}
Female {by name}
Band {By Name}

Region – A piece sometimes carries a distinctive feature of a particular region of India which may influence whether the browser wants to listen to it or not.
Its relational schema looks as follows.
Region{North India, Central India, Western India, Deccan, South India, East India, North East India}

Version – Any musical piece can have several versions. With the possible elimination of remixes, one may have different versions of the same song while still being sung by the same artist. In that scenario, the distinguishing feature may be its version. In essence, version stamps a time of origination for the musical piece.
Its relational schema looks as follows.
Version {Cover, Original Sound track, Live, Remix}

5 An Experiential Interface – Musical 'Smell' and Musicons

As shown in the sweets analogy, the music repository will be traversed at every point after the user has experienced/ sampled a small part of the fare that lies in front of him. To provide him with this interface, we introduce the concepts of a musical 'smell' which will guide the user in his peripheral search for the audio piece of his choice. We also utilize the concept of a musicon, which is the sample that he gets to experience before making his choices.

Again taking help of another analogy, consider you are entering a room, which has certain doors which are marked according to the 8 musical piece characters we described earlier. As you go near each door, you hear a small audio piece, which gives the characteristic flavor of the music hidden behind that door. As and when you like a piece, you enter that door. By choosing any door, you select a particular value for that characteristic, as marked on the door (say by choosing the *raga* door, you have selected the *ragas* of bhairavi *thaat*). Once you enter the room, which lies behind this door, you get 6 doors, marked by those characteristics, which are still left. And in this way you go round till you have identified the particular piece you want. The pieces that are played at every door of entry, which serve as an icon for that particular characteristic, are called musicons. A musicon is thus a short composition, which will give the greatest essence of the 'door' it represents. Overall, one may choose to have the most common subset of pieces to satisfy all the different doors. So a piece may be the best fit as a *dadra* (a form of art music) example. Similarly it may be the best fit for a vocal piece played along with a *pakhawaz*. Selecting the correct musicons proves to be a major challenge of this approach, since if the most 'enticing' piece is not provided, the user may very well not elect entering that 'door'. But for different users, the requirements may be different. Hence what may entice a connoisseur may prove to be too obtuse for a first time listener. To arrange for the best possible musicon, one has thus to get some information about the particular user. His basic requirements may be encapsulated in some simple information that he may be asked to provide. Based on the information that he provides, a piece is dynamically chosen as the musicon for a particular 'door', provided that he chooses to 'test' that door.

The 2nd concept of musical 'smell' is a modification of the ideas propagated in the focus with context [2] presentation as can be seen in practice in [3]. Information scent was initially proposed in the information foraging theory and has been developed after over 20 years of research work at Xerox parc on how people perceive, navigate and assimilate information. Information scent is provided by the proximal cues perceived by the user that indicates the value, cost of access, and location of distal content. In the context of foraging for information on the World Wide Web, for example, information scent is often provided by the snippets of text and graphics that surround links to other pages [4]. In our case, the aural scent will be hidden in the pieces that are selected as musicons. Basically they will be subtle pointers to the sister 'doors' that one may traverse from that particular 'door'. So if one has chosen to listen to a *Bageshri raga* rendition, the piece selected would also bear characteristics of the *Kafi thaat,* which contains the *Bageshri raga,* thus providing a musical 'smell' about say, the *Bahar raga,* which is another *raga* of the *Kafi thaat.* In this way, there will be a network of interconnections between sister nodes in the framework and the user will be provided a choice to make a horizontal detour instead of always going in a relative top-down traversal.

6 Dynamic Generation of the Interface

One of the major challenges is to dynamically generate the links, which would facilitate the quickest traversal for the user while still maintaining the exploratory nature of the interface. For example a sample traversal of the database is given below in fig. 1. At each level of the traversal, the user is faced with certain choices, and .

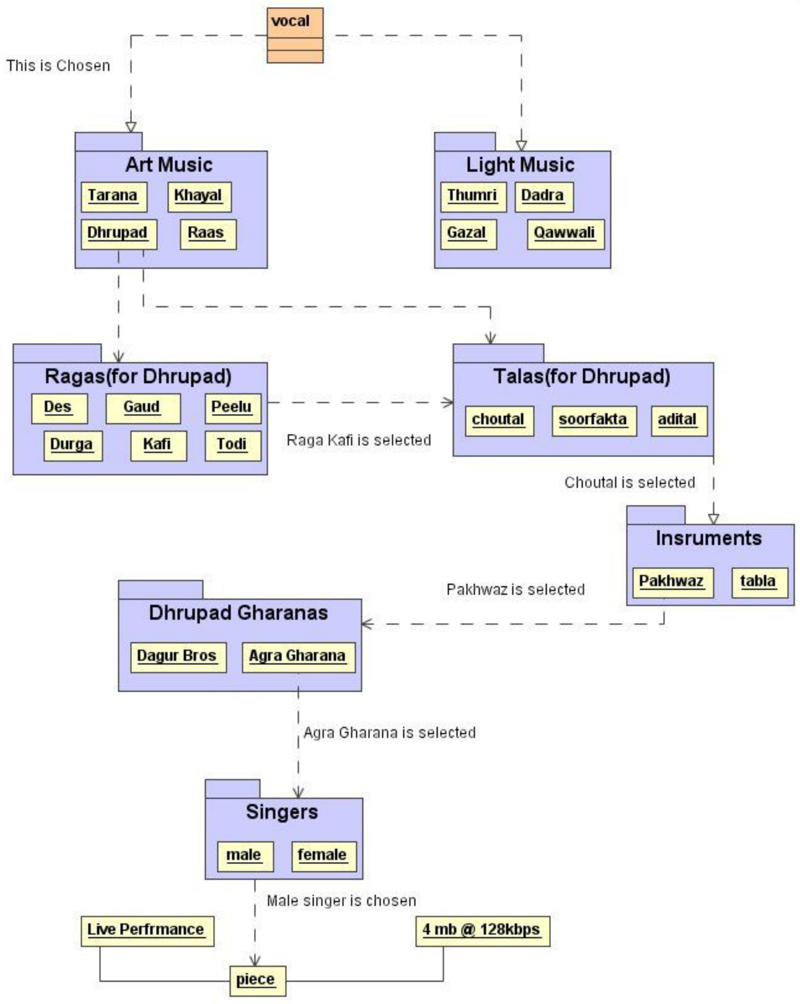


Fig. 1. Traversal path where a user selects a piece of Dhurpad music sung in Kafi raga set to choutal tala, performed by a male singer to the accompaniment of the pakhwaz. The user starts off from the node vocal and ends in the piece.

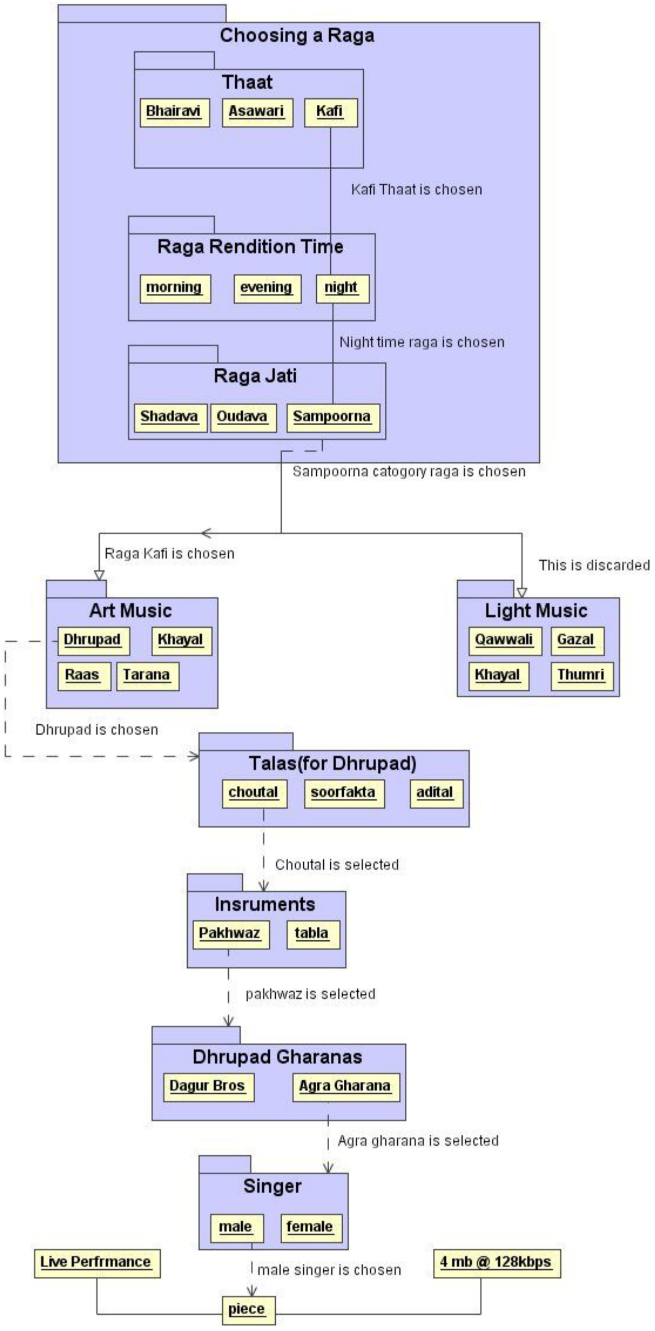


Fig. 2. The traversal path where the user selects the same piece as earlier but starting off from raga node this time. Selecting the raga involves either naming it outright or giving details of raga class so that the set is narrowed down.

based on the choice he makes; he is guided to the piece of his liking. All other choices at the same level are suppressed once a user has made a choice. But this suppression is not permanent since the user always has the option of retracing his track and coming back the way he went and then go on to a sister traversal route. In fig. 2 we can see an alternative part that a different user has used to arrive at the same piece. So the path is only a function of the choices that the user makes, based on the musicons he gets to experience.

The point to be appreciated in this is that one may browse through a very large database in a few steps and also be able to dynamically generate/permit multiple traversal mechanisms. Following is a sequence diagram in UML for the dynamic generation of the traversal routes.

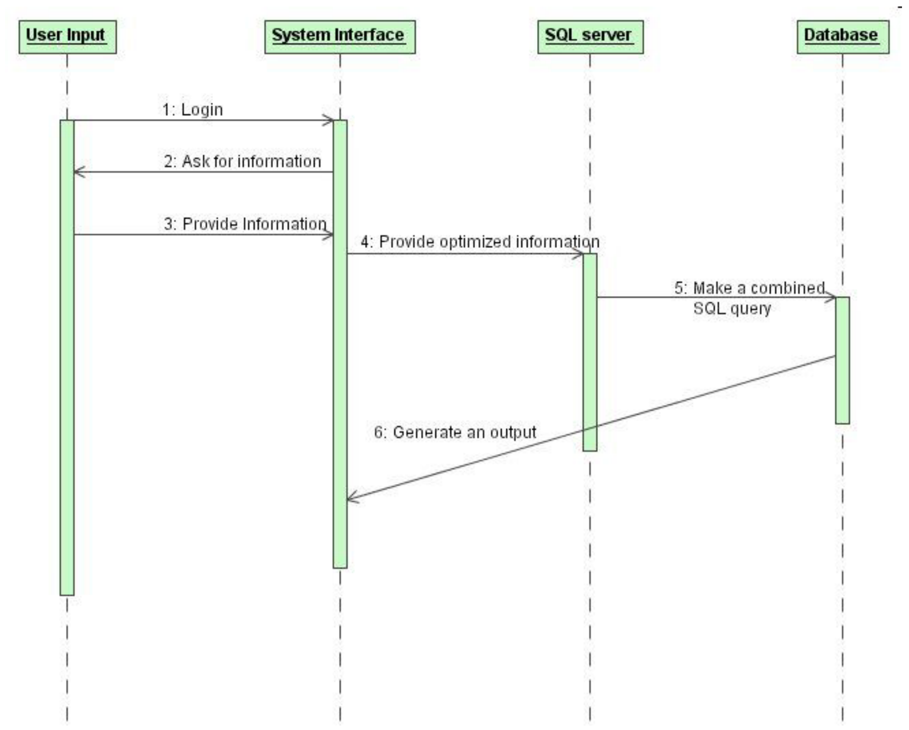


Fig. 3. Dynamic generation of the traversal route

Our traversal scheme is completely oriented according to the user’s wish. However in case the user doesn’t specify a choice, we propose a default-optimized traversal which will lead the user to his piece of liking in swiftest possible time. For this default traversal, we need to keep in mind an optimization scheme, which will make the length of traversal minimum while still retaining our original motive of giving the user the greatest possible number of choices. For optimization purposes, we have ranked each of the 7 characteristics and then their own sub divisions according to the number and size of partitions that they divide the superset into. They are in order of decreasing factoring potential of the superset.

Artist Name - 1
Type - 2
Raga - 3
Thaat - 3.1
Time of singing - 3.2
Jati - 3.3
Version - 4
Tala - 5
Gharana-6
Region – 7

Fig. 4. Gives ranking scheme for each characteristic in decreasing order of their potential to divide the subset

Here Thaat, Time of singing, and Jati are used to subdivide the raga. They are separately mentioned because they have different segmentation potentials as opposed to say artist which can be divided into male, female and band – each with equal probability.

The algorithm that is used to generate the dynamic traversal route can be formulated as below.

By sub-characteristics we mean the divisions of a characteristic so the sub-characteristics of "type" are "vocal", "instrumental" & "jugalbandi" and so on.

So it is finally simplified to the level of a series of selections. The user gets the opportunity to choose the relation on which he wants to perform the selection. The whole query is a succession of selection operations.

7 Some Critical Challenges Overcome

Any experiential interface has the advantage of favoring the user's cognitive space while making a traversal. More often than not, the repositories are places from which someone needs to buy something. An experiential network improves the chances of a product being satisfactorily selected multiple times as against a normal static network.

The dynamic generation of the path makes it ideal for a swift traversal. Since the path is generated keeping in mind the specific user requirements, one gets a shortest mean path between the starting node and the destination node, while still exploring the major hop points. In essence these are not discrete jumps to minimize the time of traversal only, they form a continuum in the musicological exploration space so as it is a smooth curve from the initial point to the destination. This is especially important since one is also trying to promote Indian classical music as an entity and wouldn't want to miss out on an admirer, for lack of presenting to him, that aspect of Indian classical music. However, exceptions can be made for this in case the user specifically wants to directly go to a certain piece without any intermediate hops. In such cases, the user will usually be a connoisseur since he will know the exact characteristics of the piece that he wants and has made a selection prior to coming to the repository.

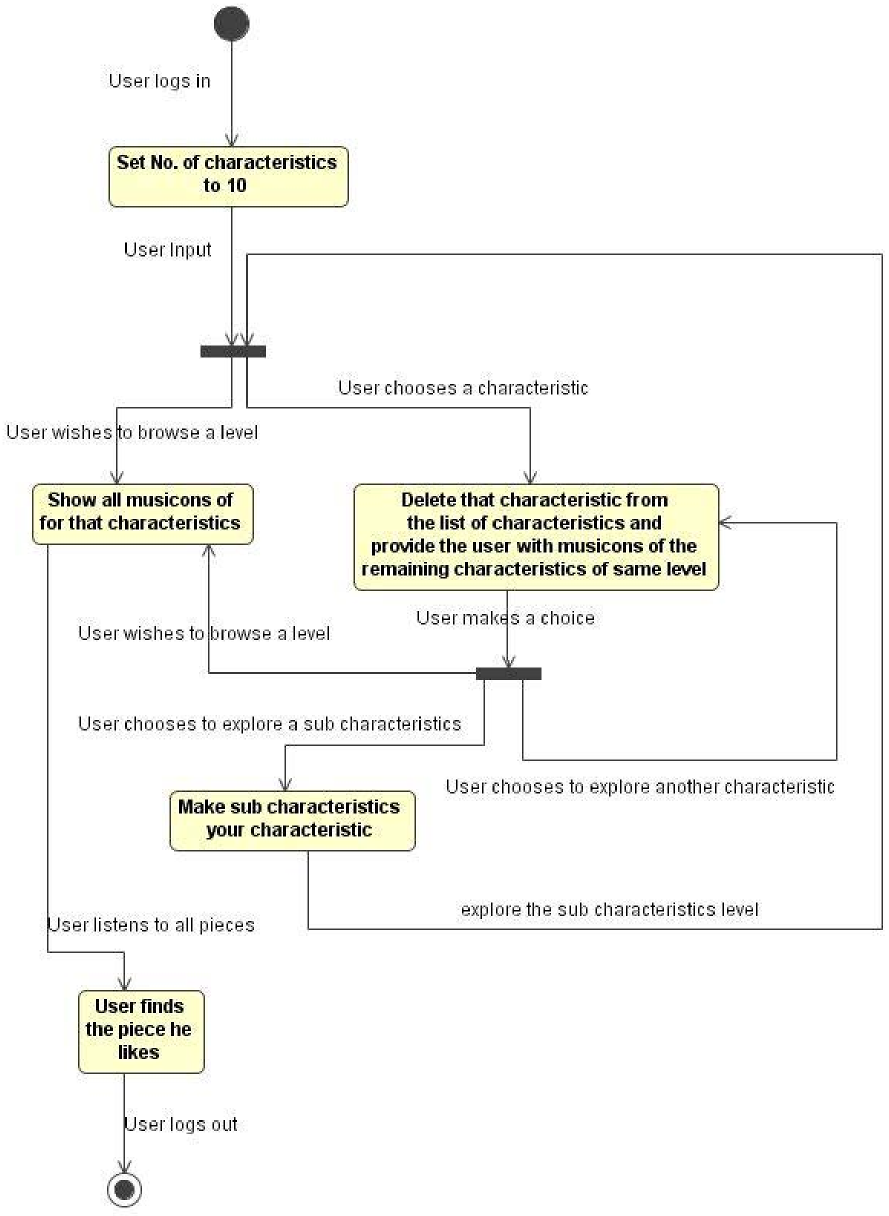

User logs in
Set No. of characteristics to 10
User Input
User chooses a characteristic
User wishes to browse a level
Show all musicons of for that characteristics
Delete that characteristic from the list of characteristics and provide the user with musicons of the remaining characteristics of same level
User makes a choice
User wishes to browse a level
User chooses to explore a sub characteristics
User chooses to explore another characteristic
Make sub characteristics your characteristic
explore the sub characteristics level
User listens to all pieces
User finds the piece he likes
User logs out

Fig. 5. Algorithm for dynamic generation of traversal path after processing user input

The whole theme of a user-modeled path gives the user full liberty in satisfying her particular requirement. He may thus decide to explore the various facets of Indian classical music by wandering through the different 'doors' or he may choose to eliminate the 'doors' one by one and thus arrive at a unique piece. He may also backtrack on the 'doors' and thus make detours or circumvent some particular aspect.

The choice of musicon assumes great importance given the experiential nature of the network. The musicon should be musically representative of the collection it stands for. While it may not be difficult to create musicons to differentiate a collection of percussion musical pieces from string instrument pieces, having a musicon to separate evening music from the afternoon may be more difficult. Also the ability to design musicons to represent the size of the collection, popularity, cost and other such properties will be very interesting.

8 Current Status

Presently we have solved the intricacies of the conceptual problem and are on way to implementing the back-end and the front-end user interface. The other major thing that is underway is choosing and collecting the most appropriate musicons for each category. A good list of musicons will bring a purely aural interface (devoid of any textual part) for the user into the realms of possibility.

9 Conclusions

Experiential framework for browsing is a very powerful model for exploring huge databases. By empowering the user at each level and generating the traversal route according to his choices, this model is ideally suited for most business level interactions. The dynamic generation of the route is another key feature of this model and it is expected to provide the thrust in time optimization of the traversal route. Appropriate design of musicons will help in making the choice and the interface could be driven mostly through an audio interface.

The technology that we have discussed for music in particular, can be extended to books, compact disks and other media which provide an experiential possibility. It may further be used for online stores, digital library, human genealogy determination etc.

References

1. Hindustani Music – Ashok Da. Ranade, National Book Trust, India
2. Rao Ramana, “See & Go Manifesto” Interaction Magazine, (Sept 1999) http://www.ramanarao.com/papers.html
3. http://www.inxight.com
4. Pirolli Peter, Card Stuart K., and Van Der Wege Mija M, "The Effect of Information Scent on Searching Information Visualizations of Large Tree Structures" (2001) http://www.inxight.com/products/vizserver/demos.php

Appendix: Musicons

The musicons are the most vital aspects of this interface since they determine the user's reaction and his next immediate step. Since it carries this cognitive burden of shaping the user's preferences, we have divided them into 3 categories: -

- For the lay user
- For users with moderate acquaintance of Indian classical music
- For a connoisseur of Indian classical music

The types of musicons will differ for each of the three categories with every piece having 3 (or more) musicons. The selection criterion for the musicons is also stringently observed, for each category of musicon acts as the virtual signature for the class of music that lies beneath it. Each musicon for each level is selected after a panel of users who classify their acquaintance with Indian classical music along those 3 different levels gives their preference. But these musicons for different categories may not be different for there may be pieces where the same musicon may serve for all three categories.

The musicons themselves will be of approximately 20-30 seconds duration. They may not be the leading part of the musical piece but will mostly be the part which catches the listener's fancy like the part which is repeated, the mukhra, a characteristic beat pattern etc. It may even have nothing to do with the piece, for example an inspired musicon for all morning ragas may be a special recording of the typical morning sounds of songbirds.

The pieces are accessible only after the user makes a definitive choice as to which piece he wants to listen to. Considering the sizes of the files, there will not be any streaming. Instead the files will be downloaded to the user's hard disk before playing. Till the user makes a choice i.e. s/he decides to download a musical piece, s/he will only hear the musicons. These musicons will thus represent the music pieces even at the lowest levels of depiction.

The traversal of the database will start from the same level i.e. after the user authenticates himself, he will find himself with the same set of 7 primary characteristics as discussed here. But once he gets beyond this primary level, he may twist the organizational structure to get to any level from any level depending upon the choice that he has made. The musicons for the topmost level will thus be a random selection since these are the 7 basic criteria on which all the pieces are classified and to play the same set of musicons for this topmost level will render the exercise meaningless. Hence a random selection of musicons according to the user cognitive level – lay user, moderate experience, connoisseur; is to be done.

Author Index