TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY
Julkaisu 817 • Publication 817

Antti Eronen

Signal Processing Methods for Audio Classification and Music Content Analysis

Tampere 2009

Tampereen teknillinen yliopisto. Julkaisu 817
Tampere University of Technology. Publication 817

Antti Eronen

Signal Processing Methods for Audio Classification and Music Content Analysis

Thesis for the degree of Doctor of Technology to be presented with due permission for public examination and criticism in Tietotalo Building, Auditorium TB109, at Tampere University of Technology, on the 25th of June 2009, at 12 noon.

Tampereen teknillinen yliopisto - Tampere University of Technology
Tampere 2009

Thesis advisor
Anssi Klapuri, Professor
Department of Signal Processing
Tampere University of Technology
Tampere, Finland

Former thesis advisor
Jaakko Astola, Professor
Department of Signal Processing
Tampere University of Technology
Tampere, Finland

Pre-examiner
Gaël Richard, Professor
Signal and Image Processing Department
TELECOM ParisTech
Paris, France

Pre-examiner and opponent
Petri Toiviainen, Professor
Department of Music
University of Jyväskylä
Jyväskylä, Finland

Opponent
Geoffroy Peeters, Ph.D.
Research and Development Department
IRCAM - CNRS
Paris, France

ISBN 978-952-15-2181-2 (printed)
ISBN 978-952-15-2196-6 (PDF)
ISSN 1459-2045

Abstract

Signal processing methods for audio classification and music content analysis are developed in this thesis. Audio classification is here understood as the process of assigning a discrete category label to an unknown recording. Two specific problems of audio classification are considered: musical instrument recognition and context recognition. In the former, the system classifies an audio recording according to the instrument, e.g. violin, flute, piano, that produced the sound. The latter task is about classifying an environment, such a car, restaurant, or library, based on its ambient audio background.

In the field of music content analysis, methods are presented for music meter analysis and chorus detection. Meter analysis methods consider the estimation of the regular pattern of strong and weak beats in a piece of music. The goal of chorus detection is to locate the chorus segment in music which is often the catchiest and most memorable part of a song. These are among the most important and readily commercially applicable content attributes that can be automatically analyzed from music signals.

For audio classification, several features and classification methods are proposed and evaluated. In musical instrument recognition, we consider methods to improve the performance of a baseline audio classification system that uses mel-frequency cepstral coefficients and their first derivatives as features, and continuous-density hidden Markov models (HMMs) for modeling the feature distributions. Two improvements are proposed to increase the performance of this baseline system. First, transforming the features to a base with maximal statistical independence using independent component analysis. Secondly, discriminative training is shown to further improve the recognition accuracy of the system.

For musical meter analysis, three methods are proposed. The first performs meter analysis jointly at three different time scales: at the temporally atomic tatum pulse level, at the tactus pulse level, which corresponds to the tempo of a piece, and at the musical measure level. The features obtained from an accent feature analyzer and a bank of comb-filter resonators are processed by a novel probabilistic model which rep-

resents primitive musical knowledge and performs joint estimation of the tatum, tactus, and measure pulses.

The second method focuses on estimating the beat and the tatum. The design goal was to keep the method computationally very efficient while retaining sufficient analysis accuracy. Simplified probabilistic modeling is proposed for beat and tatum period and phase estimation, and ensuring the continuity of the estimates. A novel phase-estimator based on adaptive comb filtering is presented. The accuracy of the method is close to the first method but with a fraction of the computational cost.

The third method for music rhythm analysis focuses on improving the accuracy in music tempo estimation. The method is based on estimating the tempo of periodicity vectors using locally weighted k-Nearest Neighbors (k-NN) regression. Regression closely relates to classification, the difference being that the goal of regression is to estimate the value of a continuous variable (the tempo), whereas in classification the value to be assigned is a discrete category label. We propose a resampling step applied to an unknown periodicity vector before finding the nearest neighbors to increase the likelihood of finding a good match from the training set. This step improves the performance of the method significantly. The tempo estimate is computed as a distance-weighted median of the nearest neighbor tempi. Experimental results show that the proposed method provides significantly better tempo estimation accuracies than three reference methods.

Finally, we describe a computationally efficient method for detecting a chorus section in popular and rock music. The method utilizes a self-dissimilarity representation that is obtained by summing two separate distance matrices calculated using the mel-frequency cepstral coefficient and pitch chroma features. This is followed by the detection of off-diagonal segments of small distance in the distance matrix. From the detected segments, an initial chorus section is selected using a scoring mechanism utilizing several heuristics, and subjected to further processing.

Keywords Audio signal analysis, audio classification, audio-based context recognition, musical instrument recognition, music meter analysis, chorus detection.

Preface

This work has been carried out at Nokia Research Center (NRC), Tampere, and at the Department of Signal Processing of Tampere University of Technology (TUT) from 1999 to 2008.

First and foremost, I wish to express my gratitude to Prof. Anssi Klapuri who introduced me to the field of audio content analysis and showed how to do high-quality research. This thesis would not exist without Anssi's invaluable contribution, professional example, and support. I also wish to thank Anssi for the fruitful collaboration between Nokia and the Department of Signal Processing of TUT. I would also like to thank my former thesis advisor Prof. Jaakko Astola for his help and support for this work, and for his contribution in creating signal processing research expertise in Tampere.

I wish to express my gratitude to the pre-examiners of this thesis, Prof. Gaël Richard and Prof. Petri Toiviainen, for their careful review of the manuscript, and to Dr. Geoffroy Peeters for agreeing to be an opponent at the public examination.

I'm very grateful to my former team leader Dr. Kari Laurila who made it possible to continue this research and supported me in finishing the thesis work. I wish to express my gratitude to our lab head Dr. Jyri Huopaniemi for participating in and supporting the audio content analysis research over many years and projects. I wish to thank my current team leader Miska Hannuksela for his support during the final stages of the thesis.

This thesis would have not been possible without the invaluable contribution of many talented people. Jarno Seppänen's meter analysis skills, software wizardism and support in general were essential for this thesis for which I'm deeply grateful. I'm grateful to all the other co-authors for their contribution for this thesis: Vesa Peltonen, Juha Tuomi, Seppo Fagerlund, Timo Sorsa, Gaëtan Lorho, and Jarmo Hiipakka. I wish to thank Timo Kosonen for his great work on the applications, Jukka Holm for helping to start the rhythm analysis research, and Mikko Heikkinen for software implementation and help. I thank Jouni Paulus for reviewing one of the papers and implementing software for another, Matti Ryynänen for his invaluable help with La-

TeX, and Mikko Parviainen for his efforts in collecting the CASR audio recordings. I'm grateful to Ole Kirkeby for reviewing the manuscript and providing valuable comments.

I wish to thank Mauri Väänänen, Jukka Saarinen, Jyri Salomaa, Jari Hagqvist and Antti Rantalahti for their facilitator work and positive attitude towards research on these topics. I want to thank Mari Mutila for her invaluable practical help. Special thanks to Matti Hämäläinen who provided me the opportunity to start working with NRC, and Jari Yli-Hietanen and Pauli Kuosmanen who provided me the opportunity to start working on audio signal processing at TUT.

I would like to extend my thanks to colleagues and friends for creating an enjoyable working environment at NRC and TUT. Especially, I would like to mention Tommi Lahti, Jussi Leppänen, Arto Lehtiniemi, Miikka Vilermo, Marko Takanen, Juha Arrasvuori, Tuomas Virtanen, Marko Helen, Konsta Koppinen, Sami Kuja-Halkola, Juuso Penttilä, Riitta Niemistö, Toni Heittola, and Timo Viitaniemi.

Funding and financial support by the Tampere Graduate School in Information Science and Engineering (TISE), TEKES, the Nokia Foundation, and Tekniikan Edistämissäätiö is gratefully acknowledged. A section in the introductory part of the thesis was written with funding from the European Commission 7th Framework Programme SAME project (no. 215749).

Kiitos vanhemmilleni tuesta ja kannustuksesta. Tämä kirja olisi varmaankin jäänyt kirjoittamatta jos en olisi tullut sitä joskus vanhemmilleni luvanneeksi.

Kaikkein suurin kiitos kuuluu vaimolleni Katrille rakkaudesta, tuesta, ja ymmärryksestä.

Tampere, May 2009. Antti Eronen

Contents

List of Included Publications

This thesis consists of the following eight publications, preceded by an introduction to the research field and a summary of the publications. Parts of this thesis have been previously published and the original publications are reprinted, by permission, from the respective copyright holders. The publications are referred in the text by [P1], [P2], and so forth.

P1 A. Eronen, A. Klapuri, "Musical Instrument Recognition Using Cepstral Coefficients and Temporal Features", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2000*, pp. 753–756, Istanbul, Turkey, June 2000.

P2 A. Eronen, "Comparison of Features for Musical Instrument Recognition", *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, WASPAA 2001*, pp. 19–22, New Paltz, New York, USA, October 2001.

P3 A. Eronen, "Musical Instrument Recognition Using ICA-Based Transform of Features and Discriminatively Trained HMMs", *Proceedings of the Seventh International Symposium on Signal Processing and Its Applications, ISSPA 2003*, Vol. 2, pp. 133–136, Paris, France, July 2003.

P4 A. Eronen, V. Peltonen, J. Tuomi, A. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, J. Huopaniemi, "Audio-Based Context Recognition", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 14, No. 1, pp. 321–329, January 2006.

P5 A. Klapuri, A. Eronen, J. Astola, "Analysis of the Meter of Acoustic Musical Signals", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 14, No. 1, pp. 342–355, January 2006.

P6 J. Seppänen, A. Eronen, J. Hiipakka, "Joint Beat & Tatum Tracking from Music Signals", *Proceedings of the 7th International Conference on Music Information Retrieval, ISMIR 2006*, Victoria, Canada, October 2006.

P7 A. Eronen, A. Klapuri, "Music Tempo Estimation Using k-NN Regression", *IEEE Transactions on Audio, Speech, and Language Processing*, accepted for publication.

P8 A. Eronen, "Chorus Detection with Combined Use of MFCC and Chroma Features and Image Processing Filters", *Proceedings of the 10th International Conference on Digital Audio Effects, DAFx-07*, Bordeaux, France, September 2007.

List of Abbreviations

ACF	Autocorrelation function
AM	Amplitude modulation
BIC	Bayesian information criterion
BOF	Bag-of-frames
BPM	Beats-per-minute
CASA	Computational auditory scene analysis
CQT	Constant-Q transform
DFT	Discrete Fourier transform
DRM	Digital rights management
EM	Expectation-maximization
ERB	Equivalent rectangular bandwidth
F0	Fundamental frequency
FIR	Finite impulse response
FFT	Fast Fourier transform
FM	Frequency modulation
GACF	Generalized autocorrelation function
GMM	Gaussian mixture model
GPS	Global positioning system
GUI	Graphical user interface
HMM	Hidden Markov model
HPCP	Harmonic pitch class profile
HWR	Half-wave rectification
IDFT	Inverse discrete Fourier transform
ICA	Independent component analysis
IIR	Infinite impulse response
IOI	Inter-Onset-Interval
k-NN	k-Nearest Neighbors
LDA	Linear discriminant analysis
LP	Linear prediction
LPC	Linear prediction coefficient
MI	Mutual information
MIDI	Musical instrument digital interface
MIR	Music information retrieval
MFCC	Mel frequency cepstral coefficient

MPEG	Moving picture experts group
OTA	Over-the-air
PC	Personal computer
PCA	Principal component analysis
PCP	Pitch class profile
PLP	Perceptual linear prediction
RMS	Root-mean-square
SC	Spectral centroid
SACF	Summary autocorrelation function
SOM	Self-organizing map
SVM	Support vector machine
ZCR	Zero crossing rate

Chapter 1

Introduction

Imagine walking on a street and listening to your favorite string quartet from the head phones of your portable music device. As you are walking towards the city center, the traffic gets harder and the noise level in the surrounding environment increases. At some point you may need to switch from classical music to something 'louder' such as heavy metal as the quiet nuances of the violin performance are barely audible anymore.

We are starting to have more and more devices that automatically adapt to the situation and adjust their behavior accordingly. In the above case, for example, the device might use its microphone to sense the increased noise level and make a decision to adjust the current playlist to incorporate music that is better audible in the loud environment. Modern hearing aids already adapt their behavior according to the environmental noise levels. As another example, consider a device which would automatically detect that the user goes jogging and select the playlist accordingly. The individual songs in the playlist could be selected to provide suitable motivation for different parts of the exercise, so that songs with slower tempo are played when the pace is slower and songs with faster tempo when running faster.

To be able to make sophisticated decisions on what music to select in each context, the system needs information on the user's context and music content. Context information may include e.g. recognizing the location, such as in a car or at home. Many sensors are available for context sensing including acceleration, illumination, global positioning system (GPS) location, temperature, camera, or microphone. Each sensor type has its own benefits regarding power consumption, cost, and type of information it provides. Context recognition using using audio is attractive since microphones are already available in many portable devices such as mobile phones, and audio provides a rich source of context information. Automatic audio content analysis methods can be used

to provide information on and categorize audio signals captured by the built-in microphone.

Music content information includes for example genre, style, release year, mood, harmony, melody, rhythm and timbre. Some of these attributes such as the genre and release year are usually available as textual metadata. By employing a number of music experts it is possible to categorize even large catalogues of music with regard to several musical attributes and use this information in making music recommendations, as is done e.g. by the personalized radio service Pandora.com. However, using human experts is costly and slow, making the development of automatic music content analysis methods attractive. Compared to human abilities, machine analysis of music content is only in its infancy. In some applications, such as tempo estimation or chorus detection from popular and rock music, machines obtain accuracies up to 90% which makes building practical applications possible. In addition, music content information such as tempo and timbre can combined with textual metadata such as genre and release year to improve the performance e.g. in content-based retrieval.

The following lists some other applications of audio content analysis.

- Multimedia information retrieval and indexing is facilitated by automatic analysis of e.g. events in a video soundtrack or attributes of a musical piece [32].

- Content modification and active listening can be enabled with content data describing the beats and measures [83]. For example, consecutive tracks can be mixed in a beat-synchronous fashion to make a smooth transition. Music player interfaces may provide novel functionality such as looping or skipping to musically meaningful locations such as the beginning of the next chorus [66].

- Music transcription means transforming an acoustic music signal into written music, a score [99]. Amateur musicians would benefit from applications which would reliably convert their favorite music collections to notated form.

- Object-based audio coding aims at using high-level objects such as musical notes as a basis for compression [174]. Being able to encode and represent sound producing objects separately would enable e.g. changing the lead instrument to something else or changing its playback style during resynthesis.

- Automatic accompaniment systems make it possible for soloists to practice with a virtual accompaniment which follows the soloist [169, 151].

1.1 Terminology

1.1.1 Musical terminology

A musical sound is often characterized with four main perceptual attributes: *pitch*, *loudness*, *duration* and *timbre*. These four attributes make it possible for a listener to distinguish musical sounds from each other. Pitch, loudness and duration are better understood than timbre and they have clear physical counterparts. For musical sounds, pitch is usually well defined and is almost equal to inverse of the period for sounds that are periodic or nearly periodic. *Fundamental frequency* F_0 is the corresponding physical term and is measured in Hertz (Hz). Pitched musical sounds usually consist of several frequency components. A perfectly *harmonic sound* with fundamental frequency F_0 has harmonics at integer multiples of the fundamental frequency.

According to Shephard, the perception of musical pitch can be graphically represented using a continually cyclic helix having two dimensions: chroma and height [164]. *Chroma* refers to the position of a musical pitch within an octave, i.e., a cycle of a helix, when seen from above. Pitch height refers to the vertical position of the helix seen from the side.

The physical counterpart of loudness is *intensity*, which is proportional to the power of an acoustic waveform. The third dimension, perceived duration, corresponds quite closely to the physical duration for tones that are not very short.

Timbre is the least understood among the four attributes. It is sometimes referred as sound ”color” and is closely related to the recognition of sound sources [71]. When two musical sounds have equal pitch, loudness and duration, timbre is the property which makes it possible to distinguish the sounds from each other. Timbre is a multidimensional concept and depends mainly on the coarse spectral energy distribution and its temporal evolution.

Musical meter relates to rhythmic aspects of music. Perceiving the meter can be characterized as a process of detecting moments of musical stress from the signal and inferring the underlying periodicities. Pulse sensations at different levels together constitute the meter [99]. The most distinct level is the one corresponding to individual beats, and is called the *beat* or *tactus*. This is the rate at which most people tend to tap their foot on the floor while listening to music. The *tempo* of a piece is defined as the rate of the tactus pulse. It is typically represented in units of beats per minute (BPM), with a typical tempo being of the order of 100 BPM. The sequence of *musical measures* relates to harmonic changes or the length of musical patterns. *Bar lines* separate the measures in musical notation. Typically, every Nth beat coincides

with the beginning of a measure. In a 4/4 *time signature* typical for Western popular music, every 4th beat coincides with the beginning of a measure, and is called a downbeat. The shortest meaningful duration encountered in music is called temporal atom or *tatum* and often coincides with the duration of 8th or 16th note.

On a larger timescale than the measure, the form of Western popular and rock music pieces often consists of distinguishable sections such as intro, verse, bridge, chorus, and outro [121]. The different sections may repeat and a typical *structure* of a musical work consists of one or more repetitions of a verse and chorus. The *chorus* is often the ”catchiest” and most memorable part of the song and is thus good to be used for music previewing, as a so-called music thumbnail [16]. Another use for the chorus section is as a mobile phone ring tone.

1.1.2 Context and metadata

Moran and Dourish define *context* as the physical and social situation in which computational devices are embedded [129]. In its general sense, context can describe the state of the environment, the user, and the device. For the purposes of this study, context describes the situation or physical location around an entity. The basic goal in context aware computing is to acquire and utilize information on the context of a device to provide better services for the user [129]. For example, a mobile phone may automatically go into a silent mode when it detects that the user sits in a meeting or in a concert.

Context information can also be used as an automatically created *metadata* for media such as music: for example when the device detects that the user is in a car and listens to music, it may automatically tag the played songs as suitable for the car environment and provide similar songs to the car environment later on [80, 162]. On a general level, metadata can be defined as data which describes data. Typical metadata for a music file includes information on the artist, composer, track and album title, genre, and beats-per-minute (BPM).

1.2 Related research fields

1.2.1 Computational auditory scene analysis, speech processing, multimedia content description, and audio fingerprinting

This thesis falls within the broad field of audio content analysis. This section briefly introduces some related research fields and provides references to more detailed overviews.

Audio content analysis is related to computational auditory scene analysis (CASA) [48, 176]. In this field, the ultimate goal is to analyze and interpret complex acoustic environments, including the recognition of overlapping sound events, and thus their sources.

Some related fields are more developed than e.g. those presented in this thesis, and can be used a source of methods and techniques. The speech and speaker recognition field is well developed although still under extensive research efforts. Many feature extraction and statistical modeling techniques used nowadays for environmental sound classification or music content analysis were first developed for speech. For overviews of speech and speaker recognition see [88, 61, 149, 148].

Query-by-example of audio is an important application for audio content analysis. Here, the goal is to find items with similar attributes from audio catalogues [72]. A special requirement in this area is to be able to efficiently compute distances between the audio samples in a database.

Audio fingerprinting, music recognition, or content-based audio identification is a well matured technology based on automatic analysis of audio content. Here, the goal is to link an unlabeled audio file to its metadata (artist, album, title) for the purposes of broadcast station monitoring, cleaning up metadata in music collections, or discovering the identity of a song heard in a bar. For overviews on audio fingerprinting see [30, 29, 175].

The multimedia description standard MPEG-7, developed by the Moving Pictures Expert Group standardizes the representation of content descriptive metadata, such as musical instrument parameters [122, 95]. Reference content analysis methods are given, but new content analysis methods can be developed to automatically produce this metadata. A more comprehensive review of audio content analysis is given in Chapter 2.

1.2.2 Music information retrieval

The field of music information retrieval (MIR) considers technologies to enable access to music collections [32]. MIR is a multidisciplinary field drawing from music perception, cognition, musicology, engineering, and computer science. The growth of research interest in the field is evident e.g. from the number of papers published in the Proceedings of the International Conference on Music Information Retrieval. The first conference was held in 2000 and the proceedings included 35 papers, whereas in 2008 the number of papers had grown to 111 [2].

Most commonly, digital music catalogues are accessed with the help of textual metadata [32]. As the metadata may be rich and descriptive, this provides efficient ways to access and find music. However, a problem is how to obtain high quality metadata for large music catalogues.

Companies such as Pandora.com ([5]) and AllMusic ([1]) use human experts to annotate descriptive terms for large catalogues of songs and are able to provide high quality search and music recommendation services. However, annotating a song e.g. at Pandora.com takes an estimated 20 to 30 minutes ([3]), which leads to large costs. Moreover, concerns raise of the consistency of metadata as large populations of people are needed to annotate collections of several million sound tracks.

An alternative for expert annotated metadata is to collect tags from users, as done by social music websites such as last.fm [4]. However, this leads to problems on how to mine high quality information from noisy tag clouds as typically users are allowed to assign whatever tags they desire for the music. There are also approaches where analysis of freeform text content on the Web is used to derive descriptions for music content. Brian Whitman describes pioneering work on this area in his thesis [179]. A more comprehensive review on music content analysis is given in Chapter 3.

1.2.3 Context awareness

Context recognition is defined as the process of automatically determining the context around a device. In addition to being a promising source of automatic metadata for music or other media types, information about the context would enable wearable devices to provide better service to users' needs, e.g., by adjusting the mode of operation accordingly. Recent overviews on context awareness can be found in [77] and [101].

Compared to image or video sensing, audio has certain distinctive characteristics [50]. Audio captures information from all directions and is more robust than video to sensor position and orientation. In addition, the nature of information is different from that provided by visual sensors. For example, what is said is better analyzed from audio but the presence of nonspeaking individuals cannot be detected. Audio can provide a rich set of information which can relate to location, activity, people, or what is being spoken [50]. The acoustic ambiance and background noise characterizes a physical location, such as inside a car, restaurant, or church. Different activities such as typing a keyboard or talking can be distinguished based on the sound they create.

1.2.4 Applications of audio-based context awareness and automatic music content analysis

Applications based on audio-based context awareness are still very much work in progress, and general environmental awareness based on audio input remains a difficult research problem. However, in some very

narrow fields commercial applications are emerging. For example, the smart alarm clock by Smart Valley Software detects the optimal moment to wake up by monitoring the quality of your sleep using the microphone of a mobile phone [6]. Modern hearing aids optimize their performance according to the noise quality of the environment [19].

Context-aware music services are at research prototype stage. For example, Lehtiniemi describes an user evaluation of a prototype context-aware music recommendation service in [109]. A high-level architecture of the service is described in [162].

Some fields of automatic music content analysis have reached sufficient maturity for practical applications. For example, the Nokia PC Suite software contains functionality to calculate the tempo from user's own music files. In professional applications, tempo analysis has existed for long. However, the analysis is not faultless and in (semi)professional applications the user may be able to fix the analysis errors e.g. by tapping the correct tempo, such as in the Music Maker music editing software by MAGIX. In amateur applications we cannot expect the user to be able to fix tempo estimation errors by tapping and work on robust tempo analysis methods is thus needed. In addition, some aspects of music meter are more difficult to analyze than others. For example, analyzing the average tempo can be done robustly, but positioning the beats or beat phase estimation is much more challenging. Estimating the bar line positions is also challenging but important for many practical applications, such as seamless beatmixing of tracks.

1.3 Scope and purpose of the thesis

This thesis considers methods for automatic content analysis of music and audio. Common to the selected methods is that they can be used for *automatic metadata generation for music*. The metadata can relate to the *content*, i.e. which instruments are used, what is the tempo of the piece, or where is the chorus section. Automatic music content descriptors provide an efficient means for automatically deriving content descriptive metadata from multimillion music track collections. Besides the actual music content, the metadata can relate to the *usage* or *context*, i.e. in which situation has the music been listened to. Examples include in a car, bus, outdoors jogging, or at home with friends. In the latter scenario, a mobile music player collects context information and automatically associates information describing the situation to the played music.

More specifically, methods are proposed to address different subproblems in music and audio content analysis. Publications [P1], [P2], [P3], and [P4] consider audio classification. In the first three publi-

cations the task is the classification of musical instruments, and [P4] considers the classification of the environment or device context based on the background sound ambiance.

Methods for musical instrument recognition have been originally proposed in [P1], [P2], and [P3]. The methods focus on classifying the instrument based on monophonic, single note recordings. The method proposed in [P1] suggests several frequency and time domain features that are useful for musical instrument recognition, and presents experiments using a hierarchical classification scheme utilizing the natural taxonomy of musical instrument families. In [P2] a very pragmatic approach is taken and an analysis is made of the efficiency of different features in the musical instrument classification task, and the problem of generalizing across different environments. Publication [P3] proposes the use of hidden Markov models with a left-right topology for instrument recognition and studies the use of linear feature transforms to transform concatenated MFCC and delta MFCC features.

Publication [P4] presents a method for recognizing the context based on audio. Similar techniques are applied as in [P3]. The paper focuses on techniques that could be used to improve the system's performance with negligible increase in the computational load at the on-line classification stage.

In music content description, the focus is on music meter analysis and chorus detection. Music meter analysis is considered in publications [P5], [P6], and [P7]. The method presented in [P5] is a complete meter analysis system capable of jointly estimating the tatum, beat (tactus), and bar level pulses in musical signals. However, when large music catalogues are processed or an algorithm should be run on an embedded device such as mobile phone, computational complexity becomes an issue. In publication [P6] a computationally very efficient method is proposed for beat tracking. The method runs faster than real-time on a mobile phone. The method presented in publication [P7] focuses on the most important subtask, tempo estimation, and significantly outperforms the previous methods in accuracy.

Finally, publication [P8] describes a method for chorus detection from music files. The method is computationally efficient while maintains sufficient accuracy for practical applications.

This research originated from the need to build a functional block into an automatic transcription system being constructed at the Department of Signal Processing at Tampere University of Technology. This work was originated by Anssi Klapuri who describes the work in more details in his Ph.D. thesis [99]. The latter part of the research has been done with Nokia Research Center, where the research is currently related to the development of a context aware mobile music service, which requires technologies for context sensing and music content

analysis [162].

1.4 Main results of the thesis

This section describes the main novel results and contributions of this thesis.

1.4.1 Publication 1

Publications [P1] to [P3] consider the problem of musical instrument recognition. In publication [P1], several features are proposed to describe each musical instrument note. A hierarchical classification scheme was implemented which utilizes the natural taxonomy of instrument families. The main results were:

- Novel features were proposed for musical instrument classification.
- Combining cepstral coefficients with other spectral and temporal features was proposed to effectively take into account both spectral and temporal information found important in human timbre perception experiments.
- Segmenting the note to attack and steady state segments and separately extracting features from both were proposed.
- The use of a manually-designed hierarchical classification taxonomy was evaluated and found not to improve the performance which contradicts with the earlier results of Martin [124].

1.4.2 Publication 2

Publication [P2] presents a detailed evaluation of several features for musical instrument recognition, and studies the problem of generalizing across different instances of the same instrument, e.g. different violin pieces played by different performers at different locations. The simulations were performed on a database larger than any study had used by that time. The main results were:

- When more than one example of an instrument is included in the evaluation, the performance of the system significantly drops. Generalizing across instruments and recording locations is identified as the key problem in instrument classification.
- The effectiveness of different features in instrument classification was analyzed.

- Different cepstral features were evaluated, and cepstral coefficients based on warped linear prediction were proposed. Mel-frequency cepstral coefficients were found to be the best choice considering classification accuracy and computational complexity.
- The effect of using one or several notes for instrument classification was tested.

1.4.3 Publication 3

In publication [P3], the use of hidden Markov models with a left-right topology for instrument note modeling is proposed. The motivation for using hidden Markov models for instrument notes is that the model may be able to learn the different spectral characteristics during the onset and steady states, removing the need for manual segmentation as was done in [P2]. In addition, the use of discriminative training and linear feature transforms to transform the catenated static and dynamic cepstral coefficients is proposed. The main results were:

- The use of left-right hidden Markov models for instrument note modeling was proposed.
- Transforming the features to a base with maximal statistical independence using independent component analysis can give an improvement of 9 percentage points in recognition accuracy in musical instrument classification.
- Discriminative training is shown to improve the performance when using models with a small number of states and component densities.
- The effect of varying the number of states and component densities in the HMMs is studied.

1.4.4 Publication 4

Publication [P4] presents a method for recognizing the context based on audio. Similar techniques are applied as in [P3]. The paper focuses on techniques that could be used to improve the system's performance with negligible increase in the computational load in the on-line classification stage. The main results were:

- Building context aware applications using audio is feasible, especially when high-level contexts are concerned.
- Discriminative training can be used to improve the accuracy when using very low-order HMMs as context models.

- Using PCA or ICA transformation of the mel-cepstral features does not significantly improve the accuracy, contrary to the case of musical instruments.
- In comparison with the human ability, the proposed system performs rather well (58% versus 69% for contexts and 82% versus 88% for high-level classes for the system and humans, respectively). Both the system and humans tend to make similar confusions mainly within the high-level categories.
- The recognition rate as a function of the test sequence length appears to converge only after about 30 to 60 s. Considering practical applications on mobile devices this poses challenges as we would like to use much less audio for performing the recognition to save energy.

1.4.5 Publication 5

Publications [P5], [P6], and [P7] present several methods for music meter analysis. Publication [P5] presents a complete meter analysis system which performs the analysis jointly at three different time scales: at the temporally atomic tatum pulse level, at the tactus pulse level, which corresponds to the tempo of a piece, and at the musical measure level. Acoustic signals from arbitrary musical genres are considered. The main results were:

- A probabilistic model representing primitive musical knowledge and capable of performing joint estimation of the tatum, tactus, and measure pulses was presented.
- The model takes into account the temporal dependencies between successive estimates and enables both causal and noncausal estimation.
- To overcome the problems of having very limited amount of training data, an approximation for the state-conditional observation likelihoods was presented.
- The transition probabilities were proposed to be modeled as a product of the prior probability of the period and a term describing the tendency of the periods to be slowly varying.
- In simulations, the method worked robustly for different types of music and improved over two state-of-the-art reference methods. The method ranked first in the ISMIR 2004 beat induction contest.

1.4.6 Publication 6

Publication [P6] presents the second method for music meter analysis, and focuses on estimating the beat and the tatum. The design goal was to keep the method computationally very efficient while retaining sufficient analysis accuracy. The paper presents a simplified back-end for beat and tatum tracking and describes its implementation on a mobile device. The main results were:

- The computationally intensive bank of comb-filter resonators was substituted with a discrete cosine transform periodicity analysis and adaptive comb filtering.
- The back-end incorporates similar primitive musicological knowledge as the method presented in [P5], but with significantly smaller computational load.
- A method based on adaptive comb filtering was proposed for beat phase estimation.
- Complexity evaluation showed that the computational cost of the method was less than 1% of the method presented in [P5] and the one by Scheirer [158]. However, it should be noted that the method [P5] was implemented as a combination of Matlab/C++, whereas the proposed method and Scheirer's method were implemented fully in C++. A real-time implementation of the method for the S60 smartphone platform was written.

1.4.7 Publication 7

The last publication ([P7]) in music meter analysis focuses on improving the performance in tempo estimation. The tempo is the most important metrical attribute in practical applications. The main results were:

- A method for measuring musical accentuation based on the chroma features was presented.
- A method for tempo estimation using locally weighted k-NN regression was presented. The method involves a resampling step which gives a significant improvement in performance.
- A method to compute the tempo estimate as a weighted median of nearest neighbor tempi was proposed.
- Experimental results show that the proposed method provides significantly better tempo estimation accuracies than three reference methods.

- The method is straightforward to implement and requires no explicit prior distribution for the tempo as the prior is implicitly included in the distribution of the k-NN training data vectors. The accuracy degrades gracefully when the size of the training data is reduced.

1.4.8 Publication 8

Publication [P8] presents a computationally efficient chorus detection method. This subproblem in music structure analysis was chosen as it seemed possible to obtain good accuracies and many potential applications exist. The main results were:

- A method for analyzing song self distance by summing the self-distance matrices based on the MFCC and chroma features was proposed.
- A scoring method for selecting the chorus section from several candidates was proposed.
- A method utilizing matched filter for refining the location of the final chorus section was proposed.
- The method provides a good chorus detection accuracy while being fast to compute.

1.5 Outline of the thesis

This thesis is organized as follows. Chapter 2 presents the relevant background information on feature extraction, classification, regression, and statistical modeling needed to understand the contents of the thesis. In addition, we discuss relevant research on musical instrument recognition, environmental audio classification, and relevant fields. Chapter 3 discusses relevant research on automatic music content analysis, focusing on music meter and music structure analysis. Chapter 4 discusses some new applications made possible by automatic audio content analysis techniques. Finally, Chapter 5 summarizes the observations made in this study and suggests some directions for future work.

Chapter 2

Audio classification

This Chapter provides the necessary background for audio classification and serves as an overview for publications [P1], [P2], [P3], and [P4]. We first discuss methods for feature extraction and classification, and conclude with a sections summarizing relevant research on these fields.

2.1 Overview

Figure 2.1 presents a block diagram of the main components of a generic audio classification system. The preprocessing stage consists of operations such as mean removal and scaling the amplitude to a fixed range, such as between -1 and 1. The feature extraction stage transforms the input signal into a low-dimensional representation which contains the information necessary for the classification or content analysis task. In practise, however, they also contain extra information since it is difficult to focus only on a single aspect of audio [32]. Model training either stores the feature vectors corresponding to the class of the labeled input signal as a finite number of templates, or trains a probabilistic model based on the observations of the class. In the classification step, the feature stream of the input signal is compared to the stored templates, or a likelihood value is calculated based on the probabilistic models of the trained classes. The recognition result is given as the class giving the best match. The following sections examine the techniques needed in different parts of this general system in more detail.

2.2 Feature extraction and transformation

2.2.1 Features

In this part, a selection of acoustic features for audio classification and music content analysis are presented.

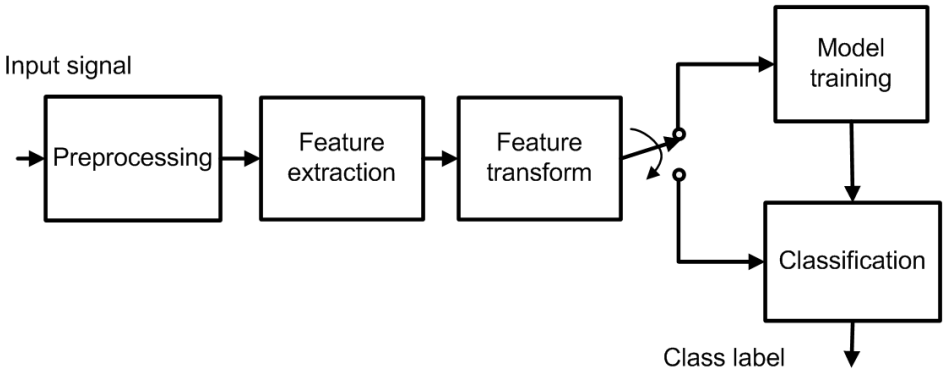


Figure 2.1: A block diagram of a generic audio classification system.

Mel-frequency cepstral coefficients

Mel-frequency cepstral coefficients ([40, 148]) and their time derivatives are the de-facto front-end feature-extraction method in automatic speech recognition systems. They have also become the first choice when building music or general audio content analysis systems. We will use here the conventional Discrete Fourier Transform (DFT)-based method utilizing a mel-scaling filterbank. Figure 2.2 shows a block diagram of the MFCC feature extractor. The input signal may be first pre-emphasized to flatten the spectrum. Pre-emphasis is typically used in speech and speaker recognition systems; for other types of signals such as environmental sounds or music it may not always be helpful. Next, a filterbank consisting of triangular filters spaced uniformly across the mel-frequency scale and their heights scaled to unity, is simulated. The mel-scale is given by

$$Mel(f) = 2595 \log_{10}(1 + \frac{f}{700}), \tag{2.1}$$

where f is the linear frequency value in Hz [148]. The mel-scale is a perceptually motivated frequency scale. It is approximately linear up to 1000 Hz and logarithmic thereafter. To implement this filterbank, a window of audio data is transformed using the DFT, and its power spectrum is calculated by squaring the absolute values of DFT output. By multiplying the power spectrum with each triangular filter and summing the values at each channel, a spectral energy value for each channel is obtained. The dynamic range of the spectrum is compressed by taking a logarithm of the energy at each filterbank channel. Finally, cepstral coefficients are computed by applying a discrete cosine transform (DCT) to the log filterbank energies. DCT decorrelates the cepstral coefficients, thereby making it possible to use diagonal covariance matrices in the statistical modeling of the feature observations.

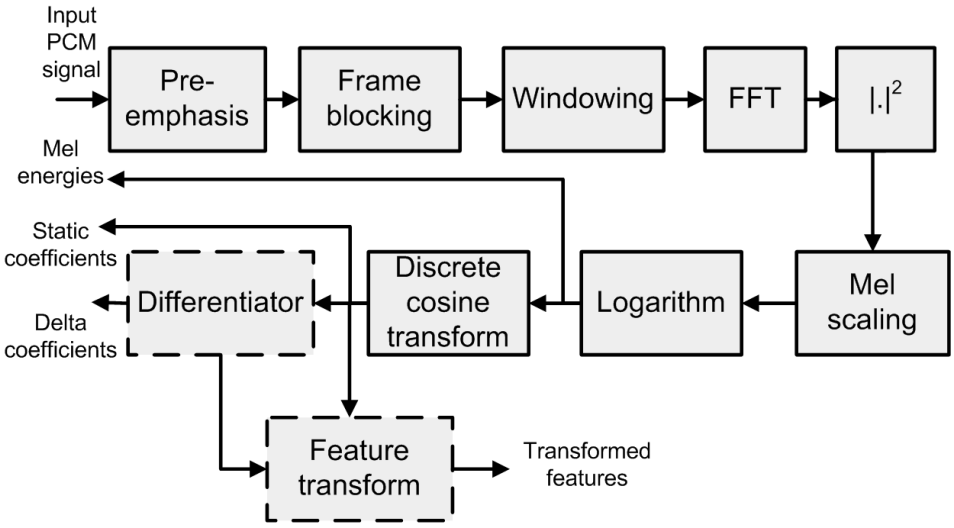


Figure 2.2: Block diagram of the MFCC analysis. Optional or new blocks are denoted with dashed lines.

In most cases, it is possible to retain only the lower order cepstral coefficients to obtain a more compact representation. The lower coefficients describe the overall spectral shape, whereas pitch and spectral fine structure information is included in higher coefficients. The zeroth cepstral coefficient is normally discarded, as it depends on the signal gain, and often we wish to ignore gain differences. The dynamic, or transitional properties of the overall spectral envelope can be characterized with delta cepstral coefficients [167, 149]. Usually the time derivative is obtained by polynomial approximation over a finite segment of the coefficient trajectory.

Linear prediction

Linear prediction (LP) analysis is another way to obtain a smooth approximation of the sound spectrum. Here, the spectrum is modeled with an all-pole function, which concentrates on spectral peaks. Linear prediction is particularly suitable for speech signals, but can be applied also to other sound source recognition tasks. Schmid applied LP analysis to musical instrument recognition already in 1977 [159].

In classical forward linear prediction, an estimate for the next sample of a linear, discrete-time system, is obtained as a linear combination of p previous output samples:

$$\widehat{y}(n) = \sum_{i=1}^{p} a_i y(n-i), \tag{2.2}$$

where a_i are the predictor coefficients, or linear prediction coefficients. They are fixed coefficients of a predictor all-pole filter, whose transfer function is

$$H(z) = \frac{1}{A(z)} = \frac{1}{1 - \sum_{i=1}^{p} a_i z^{-i}}. \tag{2.3}$$

The set of predictor coefficients $\{a_1, a_2, ..., a_p\}$ can be solved using the autocorrelation method [149]. The linear prediction cepstral coefficients can be efficiently calculated from the linear prediction coefficients using the recursion

$$c_n = -a_n - \frac{1}{n} \sum_{k=1}^{n-1} k c_k a_{n-k} \tag{2.4}$$

for $n > 0$, where $a_0 = 1$ and $a_k = 0$ for $k > p$.

The conventional LP-analysis suffers from a uniform frequency resolution. Especially in wideband audio applications, poles are wasted to the higher frequencies [79]. The technique of warped linear prediction was first proposed by Strube in 1980 [168]. In wideband audio coding, WLP has proved out to outperform conventional LP based codecs especially with low analysis orders [79]. Motivated by this, in [P2] we proposed to use cepstral coefficients based on linear prediction on a warped frequency scale. The frequency warping transform was obtained by replacing the unit delays of a discrete, linear system with first-order all-pass elements. In practice, we used the WarpTB toolbox by Härmä and Karjalainen for implementing the warped linear prediction calculation [78]. It consists of Matlab and C implementations of the basic functions, such as the warped autocorrelation calculation. The cepstral coefficients were calculated from the warped linear prediction coefficients using the recursion 2.4.

Other instantaneous features

Spectral centroid (SC) is a simple but useful feature. The spectral centroid correlates with the subjective qualities of ”brightness” or ”sharpness”. It can be calculated from different mid-level representations, commonly it is defined as the first moment with respect to frequency in a magnitude spectrum. Let $X_t(k)$ be the be the kth frequency sample of the discrete Fourier transform of the tth frame. The spectral centroid at frame t is computed as

$$SC_t = \frac{\sum_{k=0}^{K} k|X_t(k)|}{\sum_{k=0}^{K} |X_t(k)|}, \tag{2.5}$$

where K is the index of the highest frequency sample.

Zero crossing rate (ZCR) is defined as the number of zero-voltage crossings within a frame.

Short-time average energy is the energy of a frame, and is computed as the sum of squared amplitudes within a frame.

Band-energy. The band-energy at the ith band at frame t is computed as

$$BE_t(i) = \frac{\sum_{l \in S_i} |X_t(l)|^2}{\sum_{k=0}^{K} |X_t(k)|^2} \tag{2.6}$$

where S_i denotes the set of power spectrum samples belonging to the ith frequency band. The number of subbands can be defined according to the application. In [P4] we experimented with 4 and 10 logarithmically-distributed subbands.

Bandwidth measures the width of the range of frequencies the input signal occupies. In publication [P4], bandwidth is calculated as

$$BW_t = \sqrt{\frac{\sum_{k=0}^{K}(k - SC_t)^2 \cdot |X_t(k)|^2}{\sum_{k=0}^{K} |X_t(k)|^2}} \tag{2.7}$$

where SC_t is the spectral centroid measured at the frame t.

Spectral roll-off measures the frequency below which a certain amount of spectral energy resides. It measures the "skewness" of the spectral shape. It is calculated as

$$SR_t = \arg\max_p [\sum_{m=0}^{p} |X_t(m)|^2 \leq TH \cdot \sum_{k=0}^{K} |X_t(k)^2|] \tag{2.8}$$

where TH is a threshold between 0 and 1. In our experiments, the value used was 0.93.

Spectral flux (SF) measures the change in the shape of the magnitude spectrum by calculating the difference between magnitude spectra of successive frames. The spectral flux is calculated as

$$SF_t = \sum_{k=0}^{K} ||X_t(k)| - |X_{t-1}(k)||. \tag{2.9}$$

Before the low level features are fed to a classifier, certain normalizations may be applied. Especially when several different features are concatenated to a single vector, it is necessary to normalize the mean and variance using global estimates measured over the training data. This makes the contribution of different features equal. The input to the classifier is a sequence of feature vectors $\mathbf{x}_t$, where t is the frame index, and where the components of $\mathbf{x}_t$ are the values of different features.

Features for describing musical instrument notes

The previous features are instantaneous, meaning that they can be extracted from short frames of the input signal. When isolated notes

are considered, there are features that can characterize the note as a whole. The amplitude envelope of a note contains information for instance about the type of excitation; e.g. whether a violin has been bowed or plucked. Tight coupling between the instrument excitation and resonance structure is indicated by short onset durations. The amplitude envelope of a sound can be calculated by half-wave rectification and low-pass filtering of the signal. Another means is the calculation of the short time root-mean-square (RMS) energy of the signal, which we found to be a more straightforward way of obtaining a smooth estimate of the amplitude envelope of a signal. Features such as onset duration, decay-time, strength and frequency of amplitude modulation, crest factor, and detection of exponential decay can be analyzed from an RMS-energy curve. We calculated the RMS energy curve in 50% overlapping 10 ms long hanning-windowed frames.

Onset duration is traditionally defined as the time interval between the onset and the instant of maximal amplitude of a sound. *Decay time* is correspondingly the time it takes the sound to decay a certain amount, e.g. -10dB from a level corresponding to -3dB of the maximum. To measure the *slope of amplitude decay* after the onset, in publications [P1] and [P2] we proposed a method where a line is fitted into the amplitude envelope on a logarithmic scale. The fitting was done for the segment of the energy envelope that was between the maximum and the -10 dB point after that. Also, the mean square error of that fit is used as a feature describing exponential decay. Crest factor, i.e. the maximum of amplitude envelope divided by the RMS level of the amplitude envelope is also used to characterize the shape of the amplitude envelope. These three features aim at discriminating between the pizzicato and sustained tones: the former ones decay exponentially, and have a higher crest factor than sustained tones. Figure 2.3 depicts two example amplitude envelopes and the line fit used for feature extraction.

The RMS-energy envelope, now on a linear scale, can also be used to extract features measuring amplitude modulation (AM) properties. Strength, frequency, and heuristic strength (term used by Martin [124]) of amplitude modulation is measured at two frequency ranges. Rates from 4 to 8 Hz measure tremolo, i.e. AM in conjunction with vibrato, and rates between 10-40 Hz correspond to ”graininess” or ”roughness” of the tone. The RMS-energy envelope is first windowed with a hanning window. Then, FFT analysis is performed on the windowed envelope, and maxima are searched from the two frequency ranges. The frequency of AM is the frequency of the maximum peak. The amplitude features are calculated as the difference of the peak amplitude and the average amplitude, and the heuristic amplitude is calculated as the difference of the peak amplitude and the average amplitude of the frequency range under consideration.

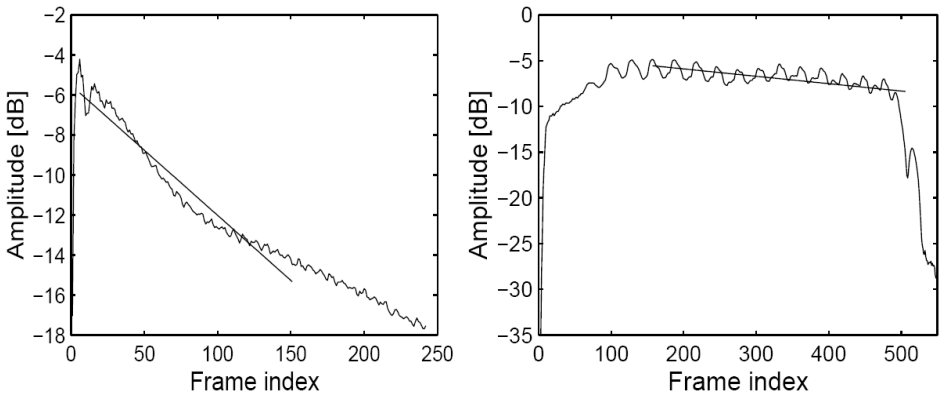


Figure 2.3: Short-time RMS-energy envelopes for guitar (left) and violin tones (right). Post-onset decay is measured by fitting a line on dB-scale. The different onset durations, slight beating in the guitar tone, and amplitude modulation in the violin tone are clearly visible.

Onset asynchrony refers to the differences in the rate of energy development of different frequency components. In [P1] and [P2] we used a ”sinusoid envelope” representation (see details in [51]) to calculate the intensity envelopes for different harmonics, and the standard deviation of onset durations for different harmonics is used as one feature. See Figure 2.4 for a depiction of sinusoid envelope representations calculated for a flute and clarinet sounds. For the other feature measuring this property, the intensity envelopes of individual harmonics were fitted into the overall intensity envelope during the onset period, and the average mean square error of those fits was used as feature. A similar measure was calculated for the rest of the waveform. The last feature calculated is the overall variation of intensities at each band. These features suffer from the difficulty of obtaining a robust representation for the development of individual partials of a tone. The sinusoidal envelope depends on obtaining an accurate estimate of the fundamental frequency and the sounds to be perfectly harmonic which is not the case for real musical instruments. A better approach would be e.g. to use a filterbank to decompose the signal into individual partials.

2.2.2 Feature transformations

The main idea of linear data-driven feature-transformations is to project the original feature space into a space with lower dimensionality and more feasible statistical properties, such as uncorrelatedness. We tested the effectiveness of some feature transformations in publications [P3] and [P4]. In order to obtain the transform matrix $\mathbf{W}$, the features

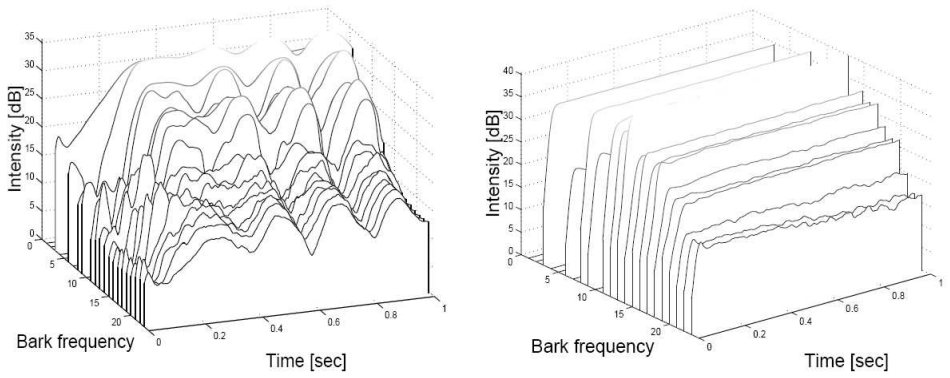


Figure 2.4: Sinusoid envelope representations for flute (left) and clarinet (right), playing the note C4, 261 Hz. Reprinted from [P1]. ©2000 IEEE.

extracted from the training data samples of all classes were gathered into a matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T]$ where each column represents the n-dimensional feature vector measured in an analysis frame. The scalar T denotes the total amount of feature vectors from all recordings of all the classes in the training set. The transform matrix $\mathbf{W}$ of size $m \times n$ is applied on $\mathbf{X}$ producing the transformed $m \times T$ dimensional observation space $\mathbf{O} = \mathbf{WX}$. In this work, three different techniques were used. The principal component analysis (PCA) finds a decorrelating transform ([44, p. 115]), independent component analysis (ICA) results in a base with statistical independence ([82][44, p. 570]), which is a much stronger condition than uncorrelatedness, and the linear discriminant analysis (LDA) tries to maximize class separability ([44, p. 120]).

Principal component analysis

Principal component analysis projects the original data into a lower-dimensional space such that the reconstruction error is as small as possible, measured as the mean-square error between the data vectors in the original space and in the projection space. The rows of the transform matrix consist of the m eigenvectors corresponding to the m largest eigenvalues of the covariance matrix of the training data. Projection onto a lower-dimensional space reduces the amount of parameters to be estimated in the classifier training stage, and uncorrelated features are efficiently modeled with diagonal-covariance Gaussians.

Independent component analysis

The goal of independent component analysis is to find directions of minimum mutual information[1], i.e., to extract a set of statistically independent vectors from the training data $\mathbf{X}$. Statistical independence is a stronger condition than uncorrelatedness. Whereas PCA results in uncorrelated variables whose covariance is zero, ICA methods consider also higher-order statistics, i.e., information not contained in the covariance matrix [82, p. 10]. The linear ICA assumes that linear mixing of a set of independent sources generates the data. More precisely, the data model is $\mathbf{x} = \mathbf{A}\mathbf{s}$, where $\mathbf{x}$ is the original feature vector, $\mathbf{A}$ is a mixing matrix, and $\mathbf{s}$ are the underlying independent sources. The goal of ICA is to estimate both $\mathbf{A}$ and $\mathbf{s}$ using the observed $\mathbf{x}$. After estimating $\mathbf{A}$, the transformation matrix is obtained as $\mathbf{W} = \mathbf{A}^{-1}$. Here, the efficient iterative FastICA algorithm was used for finding the ICA basis transformation [81].

Salam and Erten have suggested the use of ICA in context recognition by motivating that information on the movements of the user and the state of the environment is mixed in the measured signals [154]. Himberg *et al.* have used PCA and ICA to project multidimensional sensor data from different contexts into a lower dimensional representation, but reported only qualitative results [76].

In speech recognition, the use of an ICA transformation has been reported to improve the recognition accuracy [146]. In the MPEG-7 generalized audio descriptors, ICA is proposed as an optional transformation for the spectrum basis obtained with singular value decomposition to ensure maximum separability of features, and Casey's results have shown the success of this method on a wide variety of sounds [31].

There are various alternatives on how the features are input to the feature transform. In this thesis, we perform ICA on concatenated MFCC and ΔMFCC features, see Figure 2.2. Including the delta coefficients is a way to include information on temporal-dependencies of features, which is ignored if the transform is applied on static coefficients only. The results are reported in publications [P3] and [P4]. In [31] and [146] delta coefficients were not considered, and in [100] logarithmic energies and their derivatives were used. Somervuo has applied ICA on five-frame "context windows" in phoneme recognition [166].

Linear discriminant analysis

Linear discriminant analysis differs from PCA and ICA by utilizing the class labels. In this thesis, class is synonymous to an audio context or a

[1]The mutual information between two independent random variables is zero [20, p. 57]

musical instrument category. Thus, whereas PCA and ICA do not make distinction between feature vectors belonging to different classes, LDA tries to maximize the separability of data from different classes. The goal is to find basis vectors that maximize the ratio of between-class variance to within-class variance. Finding the transform matrix involves computing two covariance matrices: the within-class covariance matrix $\mathbf{S}_w$ and the between-class covariance matrix $\mathbf{S}_b$ ([44, p. 120]). The rows of the transform matrix are the m eigenvectors corresponding to the m largest eigenvalues of the matrix $\mathbf{S}_w^{-1}\mathbf{S}_b$. An additional limit for the dimension of the resulting features is presented by the fact that for C classes there are at most $C-1$ linearly independent eigenvectors ([44, p. 124]).

It should be noted that the extra computational load caused by applying any of these transformations occurs mainly in the off-line training phase. The test phase consists of computing the features in the usual way plus an additional multiplication once per analysis frame with the $m \times n$ matrix $\mathbf{W}$ derived off-line using the training data. Thus, these transforms are particularly attractive in resource-constrained context recognition applications.

2.3 Classification and acoustic modeling

2.3.1 k-Nearest Neighbors

The k-nearest-neighbors (k-NN) classifier performs a class vote among the k nearest training-data feature vectors to a point to be classified ([44, p. 182][20, p. 125]). In our implementation, the feature vectors were first decorrelated using principal component analysis and the Euclidean distance metric was used in the transformed space. When the k-NN classifier is used, it is usually not feasible to perform classification on an individual frame basis, but the information of frames is usually accumulated over a certain time period by averaging. For example, in audio-based context recognition we estimated the mean and standard deviation (std) of the features over one-second windows with an intention to model the slowly-changing attributes of environmental audio, such as finite-length acoustic events, and to reduce the computational load at the classification stage. These values were used as new feature vectors. For musical instruments, we have used e.g. averaging over the onset and steady state segments separately, and then catenating the features from the different segments into a long feature vector.

The k-NN algorithm can be applied also to regression problems. The difference is that in regression the output value to be predicted is continuous in opposite to being discrete as in classification tasks. In a typical

scenario of k-NN regression the property value of an object is assigned to be the average of the values of its k nearest neighbors. The average can also be a distance weighted average, in which case the method is an example of locally weighted learning [12]. The distance function must fulfill the following requirements: the maximum value is at zero distance, and the function decays smoothly as the distance increases [12]. In [P7] we compute the tempo as a weighted median of the nearest neighbor tempi, which increases the robustness compared to a weighted average.

2.3.2 Hidden Markov and Gaussian mixture models

A hidden Markov model (HMM) ([149, pp. 321-386]), is an effective parametric representation for a time-series of observations, such as feature vectors measured from natural sounds. In this work, HMMs are used for classification by training a HMM for each class, and by selecting the class with the largest posterior probability.

In each of our classification tasks, our acoustic data comprises a training set that consists of the recordings $O = (\mathbf{O}^1, ..., \mathbf{O}^R)$ and their associated class labels $L = (l^1, ..., l^R)$. Depending of the application, l^r can express the context where the recording has been made or the musical instrument playing on the musical excerpt r. To be more specific, $\mathbf{O}^r$ denotes the sequence of feature vectors measured from recording r. The purpose of the acoustic models is to represent the distribution of feature values in each class in this training set.

Description of a HMM

A continuous-density hidden Markov model (HMM) with N states consists of a set of parameters θ that comprises the N-by-N transition matrix, the initial state distribution, and the parameters of the state densities [88]. In the case of Gaussian mixture model (GMM) state emission densities ([148]), the state parameters consist of the weights, means and diagonal variances of the state GMMs. The possibility to model sequences of states with different statistical properties and transition probabilities between them makes intuitively sense in our applications, since sounds are dynamic phenomena. For instance, one can imagine standing next to a road, where cars are passing by. When a car approaches, its sound changes in a certain manner, and after it has passed there is a clear change in its sound due to the Doppler effect. Naturally, when no cars are passing by the sound scene is rather quiet. Hopefully, the different states in the model are able to capture the different stages, and the statistical variation between different roads, cars, and recording times is modeled to some extent by the different components in the GMM state densities.

The HMM parameters can be iteratively optimized with the Baum-Welch algorithm [149]. This algorithm iteratively finds a local maximum of the maximum likelihood (ML) objective function ([18])

$$F(\Theta) = \log p(O|L) = \sum_{r=1}^{R} \log p(\mathbf{O}^r|l^r) = \sum_{c=1}^{C} \sum_{r \in A_c} \log p(\mathbf{O}^r|c), \qquad (2.10)$$

where Θ denotes the entire parameter set of all the classes $c \in \{1, ..., C\}$, and A_c is the subset of $[1, R]$ that denotes the recordings from the class c. The optimization can be done for each class separately. The optimization starts with an initial set of values for the model parameters (the initial state distribution, transition probabilities, and parameters of the state densities), and then iteratively finds a better set of model parameters. The re-estimation equations are omitted here due to space reasons and since standard formulae were used in this thesis. See the details in [149].

In the recognition phase, an unknown recording $\mathbf{O}$ is classified using the maximum a posteriori rule:

$$\hat{c} = \arg\max_c p(c|\mathbf{O}) = \arg\max_c \frac{p(c)p(\mathbf{O}|c)}{p(\mathbf{O})}, \qquad (2.11)$$

where we used the Bayes' rule. Since $p(\mathbf{O})$ does not depend on c, and if we assume equal priors $p(c)$ for all classes, we can maximize $p(\mathbf{O}|c)$. The needed likelihoods can be efficiently computed using the forward-backward algorithm, or approximated with the likelihood of the single most likely path given by the Viterbi-algorithm [149][88].

Model initialization

Careful initialization is essential for the Baum-Welch algorithm to be able to find good model parameters. This is especially true for complex models with several states (NS) and component densities per state (NC). A useful heuristic to train models so that the amount of states and component densities is iteratively increased is the following: The models are initialized with a single Gaussian at each state. The component with the largest weight is split until the desired value of NC is obtained. Each component split is followed by a specified number of Baum-Welch iterations (e.g. 15), or until the likelihood converges. There are several ways for initializing the state means and variances. One is based on using global estimates over the whole training data of each class. E.g., for each class c a three-state HMM is initialized with means $\mu_c - 0.1\sigma_c$, μ_c, and $\mu_c + 0.1\sigma_c$, where μ_c is the mean vector computed from the training data of class c, and σ_c is the corresponding standard deviation vector. The three variances can be set equal to σ_c^2. Another method is to use the

the k-means clustering algorithm to cluster the data into as many segments as there are states in the model and estimate the initial means and variances from the cluster populations.

Sometimes it may be possible to initialize the states using various heuristics. For example, when training HMM models with a left-right topology[2] for musical instrument notes we may segment the note into as many segments as there are states in the model, and then estimate the initial state parameters from these segments. The Baum-Welch iterations are then performed during which the algorithm essentially finds the optimal segment boundaries.

In practice we need to do experimentation to determine the suitable method of initialization. Especially the k-means clustering initialization leads to models of varying quality, and often it is necessary to repeat the initialization a few times, and perform cross-validation on a validation set to determine the quality of the resulting models.

What do HMM state densities model for non-speech sounds?

To gain insight into the properties of sounds modeled by different HMM states it is useful to visually study the Viterbi segmentations after training, or in the test stage. In Figure 2.5, a three-state HMM has been trained using a recording of the sound next to a road. The top panel shows the amplitude of the signal as a function of time. The high amplitude peaks correspond to passing cars. The bottom panel shows the resulting Viterbi segmentation through the three states. The state number one models the silent periods when there are no cars passing; the second state the transition periods when a car is either approaching or getting farther, and the third state the period when the car is just passing or is very close to the recording place. A similar example with a musical sound is depicted in Figure 2.5. A three-state HMM was trained on trumpet recordings, and the segmentation is shown for a melody phrase of 15 seconds in duration. By listening it was found that state one represents high-pitched notes and pauses between notes, low-pitched notes are modeled with state three. Interestingly, state two models the initial transients.

A discriminative training algorithm

Maximum Likelihood estimation is well justified if the observations are distributed according to the assumed statistical model. In our applications, it is unlikely that a single HMM could capture all the statistical variation of the observations from an arbitrary audio environment or

[2]In a model with left-right topology, state transitions to the previous state are not allowed but the process must either proceed to the next or remain in the same state.

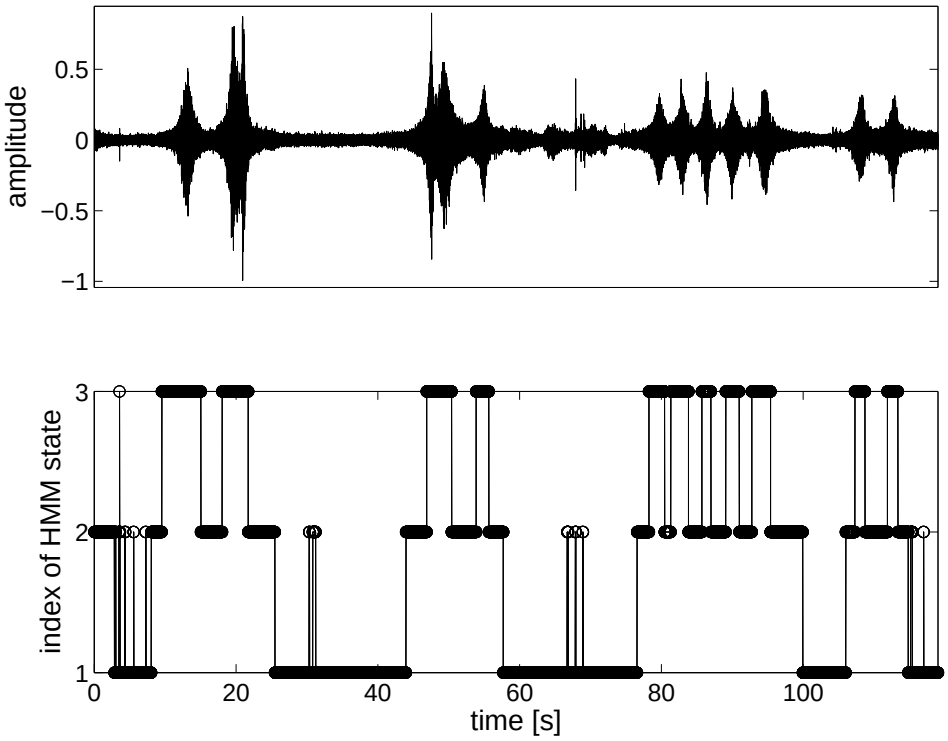


Figure 2.5: The top panel shows the amplitude of a recording made next to a road with passing cars. The bottom panel shows the Viterbi segmentation through a three-state HMM trained using the recording. The length of the analysis window is 30 ms.

all the articulation and nuances of a musical instrument, for instance. Moreover, the training databases are much smaller than for example the available speech databases, preventing the reliable estimation of parameters for complex models with high amounts of component densities. In applications where computational resources are limited such as context-awareness targeted for embedded applications, we may have to use models with as few Gaussians as possible, since their evaluation is one of the computational bottlenecks in the recognition phase. In these cases a model mismatch occurs and other approaches than ML may lead into better recognition results. Discriminative training methods such as the maximum mutual information (MMI) aim at maximizing the ability to distinguish between the observation sequences generated by the model of the correct class and those generated by models of other classes [149].

Different discriminative algorithms have been proposed in the literature. The algorithm used in this thesis has been presented recently, and one of its benefits is a straightforward implementation. The algo-

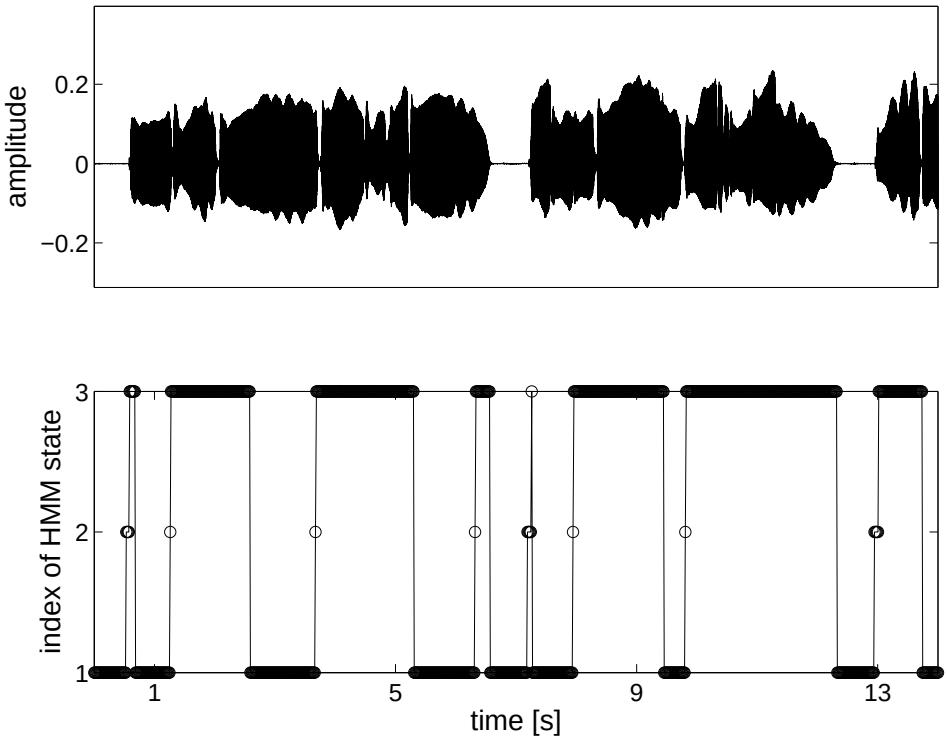


Figure 2.6: The top panel shows the amplitude of a solo melody played with a trumpet. The bottom panel shows the Viterbi segmentation through a three-state HMM trained for the trumpet class. The length of the analysis window is 30 ms.

rithm was proposed by Ben-Yishai & Burshtein, and is based on an approximation of the maximum mutual information criterion [18]. Their approximated maximum mutual information (AMMI) criterion is:

$$J(\Theta) = \sum_{c=1}^{C} \{ \sum_{r \in A_c} \log[p(c)p(\mathbf{O}^r|c)] - \lambda \sum_{r \in B_c} \log[p(c)p(\mathbf{O}^r|c)] \}, \qquad (2.12)$$

where B_c is the set of indices of training recordings that were recognized as class c. The set B_c is obtained by maximum a posteriori classification performed on the training set. The parameter $0 \leq \lambda \leq 1$ controls the "discrimination rate". The prior probabilities $p(c)$ do not affect the maximization of $J(\Theta)$, thus the maximization is equivalent to maximizing the following objective functions:

$$J_c(\Theta) = \sum_{r \in A_c} \log p(\mathbf{O}^r|c) - \lambda \sum_{r \in B_c} \log p(\mathbf{O}^r|c), \qquad (2.13)$$

for all the classes $1 \leq c \leq C$. Thus, the parameter set of each class can be

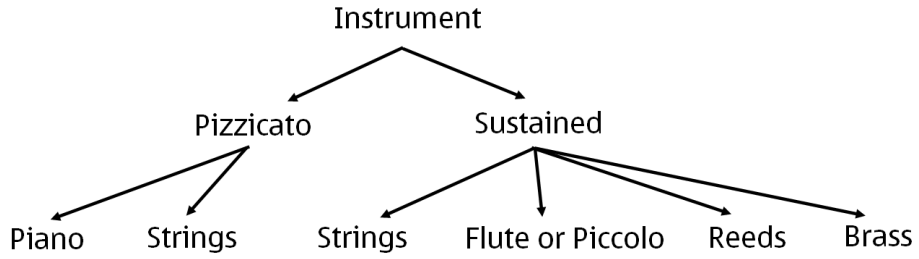


Figure 2.7: A possible taxonomy of Western orchestral instruments.

estimated separately, which leads to a straightforward implementation. The authors give the re-estimation equations for HMM parameters [18].

This discriminative re-estimation can be iterated. We used typically 5 iterations, since the improvement in recognition accuracy was only minor beyond that. In many cases, using just one iteration would be enough since it sometimes gave the greatest improvement. The recognition was done only at the first iteration, after which the set B_c stayed fixed. The following iterations still increase the AMMI objective function and increase the accuracy at least in the training set. However, according to our experience, continuing iterations too long causes the algorithm to overfit the training data, leading into poor generalization on unseen test data. Maximum of 5 iterations with $\lambda = 0.3$ was observed to give an improvement in most cases without much danger of overfitting.

2.4 Methods for musical instrument recognition

This section describes relevant research on the classification of musical instrument sounds and is background for publications [P1], [P2], and [P3].

There exists a large variety of musical instruments in the world. In practical applications, we naturally train the system with the classes of instruments that are most likely for that particular application. In this thesis, Western orchestral instruments are considered. This is done for two reasons. First, the timbre of these instruments has been extensively studied, providing insights into the information that makes recognition possible and should therefore be attempted to extract from the sounds. Second, recordings of these instruments are easily available, whereas in the cases of more exotic instruments we would first have to make the databases. Figure 2.7 presents a possible taxonomy of Western musical instruments.

In defining the musical instrument recognition task, several levels of difficulty can be found. Monophonic recognition refers to the recognition of solo music or solo notes, and is the most often studied. This study also uses isolated notes as test material mainly because samples with annotations were available with a reasonable effort, and there were published isolated note recognition systems with which the performance could be compared. However, this can be generalized to monophonic phrases by introducing a temporal segmentation stage. Polyphonic recognition has received fewer attempts. The following sections review the relevant research in these areas. For other reviews see [75, 74].

2.4.1 Monophonic recognition

Most systems have operated on isolated notes, often taken from the same, single source, and having notes over a very small pitch range. The most recent systems have operated on solo music taken from commercial recordings. The studies using isolated tones are most relevant for the results presented in publications [P1], [P2] and [P3].

Studies not testing generalization across databases

Table 2.1 presents examples of studies on classifying isolated notes on a single collection of sounds, or where examples of an instrument within the same collection may have existed both in the training and test set. As we will see later, this makes the results too optimistic. Thus, the following studies are interesting mainly from the methods point of view.

Kaminskyj and Materka used features derived from a root-mean-square (RMS) energy envelope via PCA and used a neural network or a k-nearest neighbor (k-NN) classifier to classify guitar, piano, marimba and accordion tones over a one-octave band [90]. More recently, Kaminskyj ([89]) has extended the system to recognize 19 instruments over a three-octave pitch range.

Fujinaga trained a k-NN with features extracted from 1338 spectral slices of 23 instruments playing a range of pitches [57]. A genetic algorithm was used for finding good feature combinations. When the authors added features relating to the dynamically changing spectral envelope, and velocity of spectral centroid and its variance, the accuracy improved [56]. Their latest study incorporated small refinements and added spectral irregularity and tristimulus features [58].

Martin and Kim reported a system operating on full pitch ranges of 14 instruments [125]. The best classifier was the k-NN, enhanced with the Fisher discriminant analysis to reduce the dimensions of the

Table 2.1: Summary of selected earlier research on musical instrument recognition on isolated notes with a single example of each instrument, or where the same instrument may be present in the test and train sets.

Author year ref.	Accuracy	Number of instruments
Kaminskyj 1995 [90]	98	4
Jensen 1999 [85]	100	5
Kaminskyj 2000 [89]	82	19
Fujinaga 1998 [57]	50	23
Fraser & Fujinaga 1999 [56]	64	23
Fujinaga 2000 [58]	68	23
Martin & Kim 1998 [125]	72(93)	14(5 families)
Kostek 1999 [103]	97 81	4 20
Eronen & Klapuri 2000 [P1]	80(94)	30(6 families)
Agostini *et al.* 2003 [9]	70(81)	27(6 families)
Kostek 2004 [104]	71	12
Chetry *et al.* 2005 [130]	95	11
Park & Cook 2005 [133]	71(88)	12(3 families)

data, and a hierarchical classification architecture for first recognizing the instrument families. Jensen used a Gaussian classifier and 16 parameters from his timbre model developed mainly for sound synthesis for classifying between five instruments [85].

Kostek has calculated several different features relating to the spectral shape and onset characteristics of tones taken from chromatic scales with different articulation styles [103]. A two-layer feed-forward neural network was used as a classifier. Later, Kostek and Czyzewski also tried using wavelet-analysis based features for musical instrument recognition, but their preliminary results were worse than with the earlier features [105]. In [104], Kostek reports that a combination of wavelet and MPEG-7 based features improved upon either of the features alone.

Agostini *et al.* [9] use spectral features and compare different classifiers in classifying between 27 instruments from the McGill University Master Samples collection. Support vector machines and quadratic discriminant analysis are the most successful classifiers. They report that most relevant features are inharmonicity, spectral centroid, and the energy contained in the first partial. The inharmonicity was measured as a cumulative distance between the first four estimated partials and their theoretical values.

Park and Cook extract several features from harmonic components

Table 2.2: Summary of selected research on musical instrument recognition on isolated notes across different recording conditions.

Author year ref.	Accuracy	Number of instruments
Martin 1999 [124]	39(76)	27(8 families)
Eronen 2001 [P2]	35(77)	29(6 families)
Eggink & Brown 2003 [46]	66(85)	5(2 families)
Eronen 2003 [P3]	68	7
Livshin *et al.* 2003 [115]	60(81)	8-16(3-5 families)
Peeters 2003 [137]	64(85)	23(7 families)

and use these to train a neural network classifier [133]. Their features included spectral shimmer, spectral jitter, spectral spread, spectral centroid, LPC noise, inharmonicity, attack time, harmonic slope, harmonic expansion/contraction, spectral flux shift, temporal centroid, and zero-crossing rate. Chetry *et al.* use line spectral frequencies (LSF) as features and train a codebook for each instrument using the k-means clustering method [130].

A common limitation of all these studies is that they often used only one example of each instrument, or when several databases are used, allow samples of an instrument from a database be present in both the test and training set. This significantly decreases the generalizability of the results, as we will demonstrate with our system in publication [P2], where the results are significantly worse than in [P1] where we used only samples from the McGill University Master Samples collection. Generalizing across databases is difficult.

Studies testing generalization across databases

Table 2.2 lists research which test generalization across databases. An important point is that examples of an instrument recorded in certain condition, or from a single database, are included either in the test or training set, but not both. This way, we get some evidence that the system is learning to classify an instrument (such as a violin), and not just the audio samples of a certain violin played by a certain performer in a particular acoustic place.

Martin used a wide set of features calculated from the outputs of a log-lag correlogram [124]. The classifier used was a Bayesian classifier within a taxonomic hierarchy, enhanced with context dependent feature selection and rule-one-category-out decisions.

Livshin *et al.* present an explicit test classifying instrument samples across databases [115]. It is shown how the generalization across

databases lowers the recognition accuracy. In addition, the authors report that using LDA is helpful for obtaining features that help the generalization across databases.

Peeters starts with a large set of acoustic features and then performs iterative feature selection to arrive at an optimal set of features for each part of a hierarchical classifier [137, 143]. The classifier is either k-NN or a Bayesian classifier with each class modeled as a Gaussian density. The presented results, 64% correct for 23 instruments and 85 % for families are done across databases providing a realistic estimate of the performance. The hierarchical classifiers perform better than direct classification. Although the results cannot be directly compared to our results in [P2], it is likely that his system is performing better and represents the state-of-the-art in isolated note classification. Peeters does report excluding some articulations which we did keep in our database, such as muted sounds, which probably increases the performance in his simulations. In addition, we used also synthetic notes for which the classification accuracy was very poor. However, it seems advantageous to start with a very large set of features and then perform automatic feature selection to reduce the feature set as proposed by Peeters. The feature set that was used in [P2] was smaller, and we did not fully explore the set of possible feature combinations.

Recognition of monophonic phrases

Table 2.3 presents examples of systems evaluated on monophonic phrases. On one hand, monophonic phrases are easier to classify than isolated notes as there are more than one note to be used for recognition. Publication [P2] analyzes the recognition rate as the number of notes given to the system for classification is varied. On the other hand, being able to measure onset characteristics will require a note segmentation or onset detection step, and may often be impossible when consecutive notes overlap.

Marques built a system that recognized eight instruments based on short segments of audio taken from two compact disks [123]. They used 16 mel-frequency cepstral coefficients and a support vector machine as a classifier.

Brown has used speaker recognition techniques for classifying between oboe, saxophone, flute and clarinet [26]. She used independent test and training data of varying quality taken from commercial recordings. By using bin-to-bin differences of constant-Q coefficients she obtained an accuracy of 84 %, which was comparable to the accuracy of human subjects in a listening test conducted with a subset of the samples. Other successful features in her study were cepstral coefficients and autocorrelation coefficients. In an earlier study, her system classi-

Table 2.3: Summary of selected research on musical instrument recognition on monophonic phrases.

Author year ref.	Accuracy	Number of instruments
Dubnov & Rodet 1998 [43]	not given	18
Brown 1999 [25]	94	2
Marques & Moreno 1999 [123]	83	8
Martin 1999 [124]	57(75)	27(8 families)
Brown 2001 [26]	84	4
Krishna & Shreenivas 2004 [8]	74	3
Livshin & Rodet 2004 [116]	88	7
Essid *et al.* 2006 [53]	93	10

fied between oboe and saxophone samples [25].

Krishna & Shreenivas train a GMM with line spectral frequencies (LSF) as features from individual notes of three instruments, and then classify monophonic phrases using the models [8].

Livshin and Rodet start with a very large initial set of features and then perform iterative feature selection to arrive at a feature set that classifies monophonic phrases at almost the same accuracy as the complete feature set [116].

Essid *et al.* adopt a pairwise classification strategy with GMMs or SVMs as classifiers [53]. An optimized subset of features was found for each pair of classes using a feature selection method. The authors perform pairwise classification between instrument pairs, and choose the final result as the class that wins most pairwise classifications. The authors demonstrate that the system outperforms a baseline system where a GMM is trained for each class.

Content based retrieval of instrument samples

The MPEG-7 standard presents a scheme for instrument sound description, and it was evaluated in a retrieval task as a collaboration between IRCAM (France) and IUA/UPF (Spain) in [142]. The evaluated features, or descriptors in MPEG-7 terminology, were calculated from a representation very similar to our sinusoid envelopes, which were discussed in 2.2. The authors performed an experiment, where random notes were selected from a database of sound samples, and then similar samples were searched using the descriptors, or just random selection. The subjects were asked to give a rating for the two sets of samples selected in the alternative ways. A ”mean score” of approximately 60 % was obtained using one descriptor, and approximately 80 % when using five

Table 2.4: Summary of selected research on musical instrument recognition on polyphonic material.

Author year ref.	Number of instruments	Polyphony
Eggink & Brown 2003 [46]	5	2
Livshin & Rodet 2004 [116]	n/a	2
Essid 2005 *et al.* [52]	5	max 4
Leveau 2007 *et al.* [110]	10	4
Kitahara 2007 *et al.* [97]	5	max 4
Little & Pardo 2008 [114]	4	4

descriptors.

2.4.2 Polyphonic recognition

Polyphonic instrument recognition, i.e., recognition of instruments on sound mixtures has received less research interest than monophonic instrument classification. The problem is substantially more difficult than the monophonic case. In addition to labeling the instruments, the method needs to estimate the number of instruments in the mixture. The main difficulty lies in the fact that feature extraction for each instrument is the mixture is very difficult since the harmonic partials overlap. The methods may either try to separate individual notes or instruments from the mixture and apply techniques developed for monophonic recognition, or alternatively try to extract robust features directly from the polyphonic mixture. Table 2.4 lists some approaches trying to cope with the polyphonic situation.

Godsmark and Brown used a ”timbre track” representation, in which spectral centroid was presented as a function of amplitude to segregate polyphonic music to its constituent melodic lines [60]. In assigning piano and double bass notes to their streams, the recognition rate was over 80 %. With a music piece consisting of four instruments, the piano, guitar, bass and xylophone, the recognition rate of their system decreased to about 40 %.

The work of Kashino *et al.* in music transcription involves also instrument recognition. In [93], a system transcribing random chords of clarinet, flute, piano, trumpet and violin with some success was presented. Later, Kashino and Murase have built a system that transcribes three instrument melodies [91, 92]. Using adaptive templates and contextual information, the system recognized three instruments, violin, flute and piano with 88.5 % accuracy after the pitch of the note was provided. The work was continued by Kinoshita *et.al.* [96]. The authors

presented a system that could handle two note chords with overlapping frequency components using weighted template-matching with feature significance evaluation. They reported recognition accuracies from 66 % to 75 % with chords made of notes of five instruments.

Eggink & Brown utilize the missing feature theory by marking frequency regions with overlapping partials as unreliable, assuming nearly harmonic spectra and known fundamental frequencies [46]. The features are logarithmic energies at 60Hz wide spectral bands spanning the frequency range from 50Hz to 6kHz, with 10Hz overlap between adjacent bands. Instruments are modeled with a GMM, and a binary mask is used to exclude unreliable feature components from the calculation of the GMM likelihood. A potential problem here is that the method assumes independence of feature components which does not hold for spectral energies. In the tests, the fundamental frequency was supplied to the system. The authors tested the system in a more realistic condition with analyzed F0s, but reported only preliminary results.

Essid *et al.* [52] apply their pairwise classification strategy also for recognition of polyphonic mixtures. They train pairwise classifiers between all possible instrument combinations and show promising results in recognizing typical instrument combinations for jazz music.

Leveau *et al.* [110] decompose the signal using instrument specific harmonic atoms. The authors report that classifying the instrument label without knowing the number of instruments can be done only with 17% accuracy.

Kitahara *et al.* apply linear discriminant analysis to find a feature set which is little affected by overlapping. The authors quantitatively evaluate the influence of the overlapping on each feature as the ratio of the within class variance to the between-class variance in the distribution of training data obtained from polyphonic sounds [97]. The motivation for this is the assumption that if a feature greatly suffers from the overlapping, it will have a large variation.

Livshin and Rodet report preliminary experiments on instrument recognition on duets [116]. They demonstrate that their recognizer developed for monophonic phrases performs rather well in recognizing the dominant instrument in duets when applied directly on the two-note mixtures. They also develop a system that uses an F0 estimator to find the harmonic partials in a frame, and then generate two filtered samples for recognition: one retaining only the harmonic partials and the other only the residual. The monophonic recognizer is applied separately to the samples. This latter method is more accurate in recognizing the weaker instrument.

Little and Pardo present a very interesting approach for labeling the presence of an instrument where the learning done is done on weakly labeled mixtures [114]. This means that the system is presented with

examples where only the presence of a target instrument is indicated, but the exact times during which it is active is not needed. The authors report that the system trained with weakly labeled mixtures performs better than one trained with isolated examples, and suggest that this is because the training data, in the mixed case, is more representative of the testing data, even when the training mixtures do not use the same set of instruments as the testing mixtures.

2.5 Methods for audio-based context recognition

In this section, we review some research results relevant for audio-based context recognition and especially publication [P4]. We start by briefly discussing context awareness in general without limiting to audio input only. This because the methods used for other sensory types are sometimes quite similar to those used in the audio domain, although specialized features can be developed for audio. One of the reasons for this is that since we are dealing with environmental sounds, the input can contain practically any sounds, which makes the utilization of highly specialized feature extractors a difficult task and favors generic, possibly data-driven feature extraction methods.

The second field to be reviewed is context recognition based on audio which is most relevant for us. When publication [P4] was written, there were few publications on the topic. Recently, it has started to attract increasingly more research interest. In addition, we review some results on domains which have a different problem formulation but bear similarity with regard to data or methods used. These include audio classification and retrieval, personal audio archiving, and video sound track segmentation.

2.5.1 Context awareness

In many cases the context-awareness functionality is build upon an array of different sensors sensing the context. In [106], the set of sensors included accelerometers, photodiodes, temperature sensors, touch sensors, and microphones. Low level features were then extracted from these sensor data inputs. The purpose of the feature extraction step is to transform the (often high dimensional) input data into a more compact representation while keeping sufficient amount of information for separating the different classes. As an alternative to extracting features designed using domain expertice or heuristics, blind, data driven transformations can be used. For example, principal component analysis (PCA) or independent component analysis (ICA) can be used to transform the raw input into a low dimensional representation [76, 154].

In general, the process of context recognition is very similar regardless of the sensors or data sources used for the recognition. The feature vectors obtained from sensors are fed to classifiers that try to identify the context the particular feature vectors present. As classifiers, e.g. hidden Markov models (HMMs) [35], or a combination of a self-organizing map and a Markov chain have been used [106].

2.5.2 Audio-based context awareness

Recognizing the context or environment based on audio information has started to attract increasing amount of research interest. One of the earliest studies was done by Clarkson, who classified seven contexts using spectral energies from the output of a filter bank and a HMM classifier [35]. In [155], Sawhney describes preliminary experiments with different features and classifiers in classifying between voice, traffic, subway, people, and other. The most successful system utilized frequency-band energies as features and a nearest-neighbor classifier. Kern classifies between street, restaurant, lecture, conversation, and other using a set of low-level features transformed using Linear Discriminant Analysis (LDA) and a Bayes classifier with HMM class models [94].

In publication [P4], we compared various features and classifiers in recognizing between 24 everyday contexts, such as restaurant, car, library, and office. The final system used catenated MFCCs and their first-order derivatives as features and hidden Markov model with discriminative training for classification. In addition, a listening test was made to compare the system's performance to the human abilities. The average recognition accuracy of the system was 58% against 69% obtained in the listening tests in recognizing between 24 everyday contexts. The accuracies in recognizing six high-level classes were 82% for the system and 88% for the subjects.

More recent studies have reported sometimes high performance figures with various methods and also concrete implementations on mobile devices. On a set of 27 contexts, Bonnevier has reported an accuracy of 69% with a Bayesian classifier and a subset of features obtained by running a feature selection algorithm on an initial set of MPEG-7 features, MFCCs, and zero-crossing rate [21]. Note that the method was allowed to pick individual features from a feature vector such as the MFCC which may raise concerns about overfitting the training data.

Ma *et al.* presented a HMM based environmental noise classification system and reported over 91% accuracy in ten-way classification of contexts bar, beach, bus, car, football match, launderette, lecture, office, railway station and street using three second test excerpts [120]. MFCCs augmented with the energy term and their first and second order derivatives were used as features. The authors also performed a

listening test on the same data. The listener's performance was significantly worse than the system's; this is probably due to the fact that only 3 seconds of test data was given for them. The context aware system was implemented as a client-server system where the server used an offline database to produce the noise models which were then used for online noise classification. Using the same database, Perttunen *et al.* [145] computed the averaged Mel scale spectrum over three second segments and used a Support Vector Machine (SVM) classifier and reported further improvement in the classification accuracy.

Aucouturier *et al.* have analyzed the typical Bag-of-frames (BOF) approach, where framewise features such as MFCCs are modeled with GMMs. A limitation of this approach is that the it ignores the temporal sequencing of the feature vectors: the likelihood of a feature vector sequence given the GMM parameters is the same irrespective of the temporal ordering of the feature vectors. In [13], they report on a listening test where human subjects are made to listen to "spliced" and not-spliced versions of environmental audio recordings. Spliced versions are done by splitting the audio into short frames, scrambling the order of the frames and concatenating again. The authors conclude that splicing has a significant but relatively small effect on the human performance on audio context recognition, and that the BOF approach is rather sufficient approach for audio context recognition in opposite to music similarity where the drop in recognition ability is larger. The authors also report that their result is in contradiction to our earlier study in human perception of audio environments where identification of individual sound events has been reported as a cue for identification [144].

In [14], Aucouturier *et al.* report a 90% precision in query-by-example of audio from four environmental sound classes after retrieving the five first recordings. The precision is measured as the ratio of returned recordings from the correct class to the number of retrieved recordings. The signal is modeled with MFCC coefficients and each recording with a GMM, and their distance is measured with the Kullback-Leibler (KL) divergence ([20, p. 55]) using Monte Carlo simulation. An interesting result is that, according to the authors, in environmental sounds majority of the frames are important for classification whereas in polyphonic music a minority of the frames differentiate the music from other music pieces, and majority of the frames is in fact detrimental for the performance of music similarity.

2.5.3 Audio classification and retrieval

The features typically used for audio-based context awareness are similar to those used in different audio information retrieval tasks [55]. The earliest approaches were done on classifying only a few types of envi-

ronmental noises. El-Maleh *et al.* classified five environmental noise classes (a car, street, babble, factory, and bus) using line spectral features and a Gaussian classifier [47]. Vehicle sound classification was approached using discrete hidden Markov models by Couvreur *et al.* [37]. They used linear prediction cepstral coefficients as features. The authors also described an informal listening test, which showed that, on the average, humans were inferior in classifying these categories compared to their system.

Speech/music discrimination is a typical example and the paper by Scheirer and Slaney describes a basic approach using a combination of several features [156]. In some studies environmental noise is included as one of the categories. See for example the papers by Lu *et al.* ([119]), and Li *et al.* ([113]). Various granularities of the task description are possible by further subdividing the classes. Zhang and Kuo ([181]) classified between harmonic environmental sound, non-harmonic environmental sound, environmental sound with music, pure music, song, speech with music, and pure speech. Büchler *et al.* report on classifying clean speech, speech in noise, noise, and music in hearing aids with very high accuracy except for the "speech in noise category" [19].

The MPEG-7 standard by the Moving Picture Experts Group presents methods for multimedia content description and also for describing general sound sources. Casey has used a front-end where log-spectral energies are transformed into a low-dimensional representation with singular-value decomposition and independent component analysis [31]. The proposed classifier uses single-Gaussian continuous-density HMMs with full covariance matrices trained with Bayesian maximum a-posteriori (MAP) estimation. Casey has reported impressive performance figures using the system on a database consisting e.g. of musical instrument sounds, sound effects, and animal sounds.

In a realistic audio retrieval system we need to be able to efficiently compute distances between models of audio clips in a audio database. Helen and Virtanen present various similarity measures between GMM or HMM models of features for audio retrieval of speech, music, and environmental sounds [72].

2.5.4 Analysis of video soundtracks

Analyzing and categorizing video soundtracks is a related research field to audio-based context recognition. Describing soundtracks using key audio effects is an interesting approach used for sound track categorization. In [28], Cai *et al.* propose a framework for detecting key audio effects and describing an audio scene. They use a hierarchical probabilistic model, where an HMM is first built for different audio effects based on sound samples, and then a higher level model is used to con-

nect the individual models. The optimal key effect sequence is searched through the candidate paths with the Viterbi algorithm. This approach is interesting since individual sound events have been found to be a strong cue for audio context identity [144], although the complexity of the system is likely to be too large for context awareness applications. More recently, the authors have proposed an unsupervised co-clustering approach for the same task [27].

2.5.5 Personal audio archiving

Ellis and Lee have worked on an application to record personal experience as continuous, long audio recordings [50]. Automatic analysis of the content for indexing purposes is an essential requirement as it is expected that only a fraction of the material is of any value. The authors performed automatic segmentation and labeling of 62 hours of recorded personal audio. They used the Bayesian Information Criterion (BIC) ([20, p. 216]) as a segmentation criterion, as earlier used in speaker segmentation. The distance matrix between various segments was calculated using the Kullback-Leibler divergence ([20, p. 57]) between single diagonal-covariance Gaussians fitted to the spectral features, and spectral clustering was performed on the similarity matrix to group the segments. The most successful features were average log-domain auditory spectrum, normalized entropy deviation, and mean entropy.

2.5.6 Discussion

In recent years, progress has been made in audio-based context recognition. Very good performance has been reported e.g. in [14, 120, 145], although the set of used recordings has been smaller than we have used in publication [P4]. Moreover, the database presented in [120] provides only little variation between the different recordings from the same environment and thus leads to high recognition percentages. This was tested by repeating the experiments of [120] using their publicly available data. We used a simple approach where each recording was modeled with a Gaussian fitted to its features, and classification was done with a k-Nearest Neighbor classifier with symmetrized Kullback-Leibler divergence as the distance metric. This lead to over 90% accuracy on the dataset of [120].

Chapter 3

Music content analysis

Music content analysis is a broad field covering tasks such as

- transcription of melody, bass, or chords
- analysis of meter and structure
- classification of music by genre, artist, or mood
- finding remix or cover versions of original songs.

This chapter reviews relevant research on meter and structure analysis as background for publications [P5], [P6], [P7], and [P8].

3.1 Meter analysis

Musical meter is a hierarchical structure, which consists of pulse sensations at different time scales. The most prominent level is the *tactus*, often referred as the foot tapping rate or beat. Here, we use the word

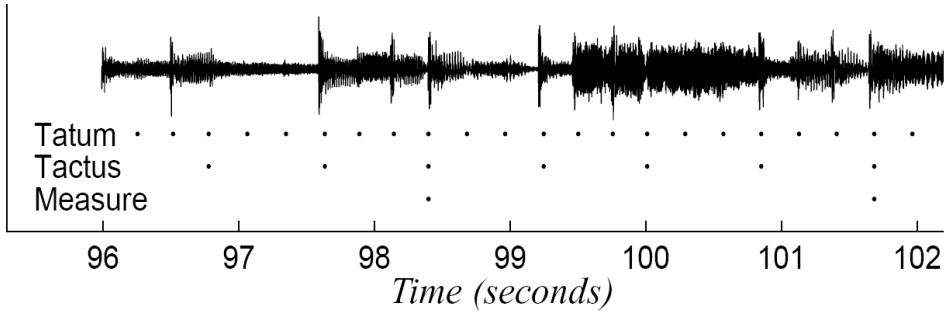


Figure 3.1: A musical signal with the tatum, tactus (beat), and measure levels illustrated. Reprinted from [P5]. ©2006 IEEE.

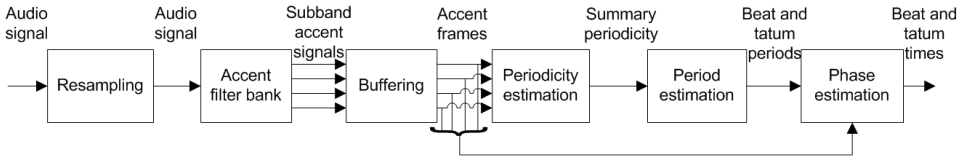


Figure 3.2: Overview of the beat and tatum analysis system of [P6], which is a good representative of the main modules in a meter analysis system. Reprinted from [P6]. ©2006 University of Victoria.

beat to refer to the individual elements that make up a pulse. Figure 3.1 illustrates a musical meter where the dots denote individual beats and each sequence of dots corresponds to a particular metrical level. We use the term *period* of a pulse to refer to the time duration between successive beats and *phase* to refer to the time when a beat occurs with respect to the beginning of a piece. The *tempo* of a piece is defined as the rate of the tactus pulse. In a musically meaningful meter, the pulse periods are slowly varying and each beat at the larger levels must coincide with a beat at the smaller levels.

3.1.1 Overview

Meter analysis involves estimating the possibly time-varying period of one or more metrical levels, and the locations of each beat. A full meter analysis system can estimate the periods and locations at the three most prominent metrical levels (measure, tactus, and tatum), whereas *beat tracking* involves estimating the time-varying tempo and the locations of the beats at the tactus level. In some applications it is sufficient to perform *tempo estimation*, i.e., to estimate the rate of the tactus pulse ignoring the phase.

Automatic rhythm analysis often entails the steps of measuring musical accentuation, analyzing the periodicity in the accent signals, and determining the period corresponding to one or more metrical levels. Figure 3.2 depicts an overview of the beat and tatum analysis system in [P6].

3.1.2 Musical accent analysis

The purpose of musical accent analysis is to extract features that correlate with the beginnings of sounds and discard information irrelevant for tempo estimation. The purpose is to device a feature that reacts to events that give emphasis to a moment in music, such as beginnings of all discrete sound events, especially the onsets of long pitched events, sudden changes in loudness or timbre, and harmonic changes. Fig-

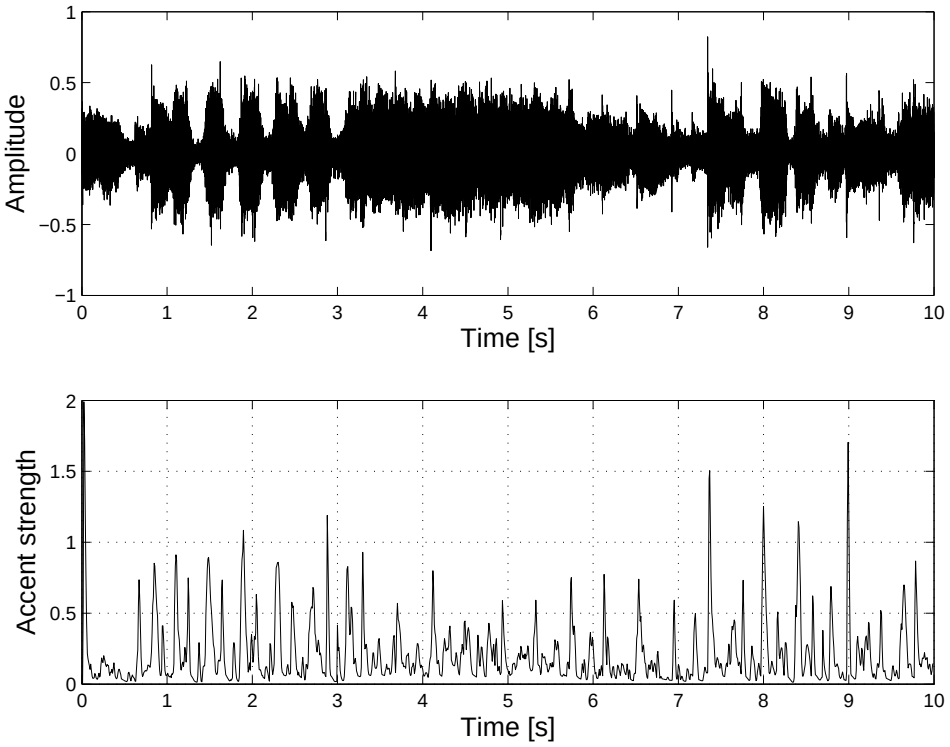


Figure 3.3: 10 second excerpt of the audio waveform of the song ”25 or 6 to 4” by Chicago (top panel), and the corresponding accent signal (lower panel).

ure 3.3 depicts an example of a musical waveform and extracted accent signal which reacts to spectral changes in the piece. Bello *et al.* divide the features used in onset detection to two broad groups: methods based on the use of signal features and methods based on probabilistic signal models [17]. The signal features include e.g. *temporal features* such as amplitude envelope, *spectral features* such as spectral difference or spectral flux, *spectral features using phase* such as the mean absolute phase deviation, and *time-frequency and time-scale analysis* based on e.g. wavelet decomposition of the signal. Another group of features is based on an assumption that the signal can be described by some probabilistic model. For example, a statistical measure of ”surprise” may consist of adapting some signal model based on incoming data, and analyzing when incoming data in a short window no longer fits the model. Another example is the log-likelihood ratio test, which entails training two probabilistic models with data on both sides of a time instant, and computing the likelihood ratio of these models.

In publications [P5], [P6], and [P7] we apply various spectral fea-

tures for musical accent analysis. The main steps in the methods are decomposing the signal into frequency bands and measuring the degree of change in the bands. The frequency decomposition can be done with the help of the DFT ([P5]), using a multirate filterbank ([P6]), or using a chroma analyzer or the mel-frequency filterbank [P7]. Chroma features will be described in more detail in section 3.2.2. In publication [P5], an accent feature extractor is presented which utilizes 36 logarithmically distributed subbands for accent measurement and then folds the results down to four bands before periodicity analysis. The benefit of using a wide range of subbands is that it is possible to detect also harmonic changes in classical or vocal music which do not have a strong beat. The method in [P6] is designed with the goal of keeping the computational cost low. The accent feature extractor based on the chroma features in [P7] can be considered to further emphasize the onsets of pitched events and harmonic changes in music. Measuring the degree of change consists of half-wave rectification (HWR) and weighted differentiation of an accent band envelope.

3.1.3 Pulse periodicity and phase analysis

Musical accent analysis is followed by periodicity analysis, since musical meter concerns the periodicity of the accent, not the onsets themselves. A natural choice is to apply a periodicity estimator, such as autocorrelation, to the accent signal to find intrinsic repetitions. The autocorrelation is defined as

$$\rho(l) = \sum_{n=0}^{N-1} a(n)a(n-l), \quad 0 \leq l \leq N-1 \tag{3.1}$$

for a frame of length N of the accent signal $a(n)$. The autocorrelation may be applied separately for a set of subbands, in which case $a(n)$ represents the accent signal from a single subband. Performing periodicity analysis directly on half-wave rectified differentials of subband power envelopes was proposed by Scheirer ([158]), and was an important advance compared to earlier methods based on discrete onset detection. Figure 3.4 depicts an example periodicity measurement from a signal using autocorrelation. Offset and scale variations have been normalized from the autocorrelation, see details in [P6]. The autocorrelation will show peaks at the lag corresponding to the basic periodicity of the accent signal, and its integer multiples.

A straightforward solution for beat or tatum period estimation consist of weighting the autocorrelation or other periodicity observation with a prior, and selecting the period corresponding to the maximum peak. This is the principle used e.g. in [P5], [P6] and [49].

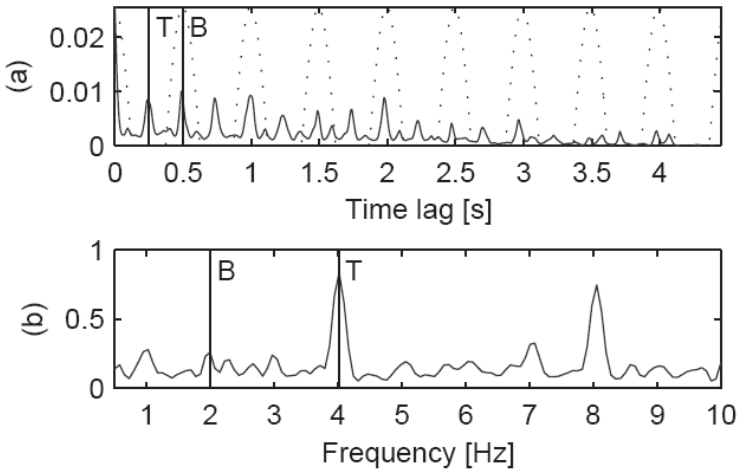


Figure 3.4: (a) autocorrelation and (b) summary periodicity, with beat (B) and tatum (T) periods shown. Reprinted from [P6]. ©2006 University of Victoria.

When an estimate of the period has been obtained, the remaining task is to position the individual beats to the timeline. This often entails making a prediction to the next beat location given the location of the previous beat and the new period estimate, and finding a local maximum of the accent signal near the predicted position. At the end of the signal, the best path through the accent signal may be searched. A good example of such a method is the dynamic programming approach presented by Ellis [49].

Some periodicity estimators provide an estimate of the phase in addition to period. Scheirer proposed the use of a bank of comb-filter resonators with constant half-time for beat tracking [158]. The accent signals are fed to a bank of comb-filter resonators with delays tuned across the range of beat periods to be measured. The energy at each band indicates the strength of periodicity in the signal corresponding to that particular delay. The delays of the comb-filter give an estimate for the beat phase. This is equivalent to using the latest τ outputs of a resonator with delay τ. The phase estimation in [P5] and [P6] is based on examining resonator outputs and defining a weight for the deviation of the phase from an ideal beat location. Ideally, the location of a beat is characterized by a large value on all accent channels, and the location does not deviate much from the ideal location obtained by adding the current period estimate to the previous beat location.

Table 3.1: Summary of selected research on music meter analysis. The values in the column Input denote A=audio, S=symbolic or MIDI.

Author year ref.	Approach	Input	Output
Allen & Dannenberg 1990 [10]	multiple agent	S	beat
Rosenthal 1992 [153]	multiple agent	S	beat, measure, time signature
Brown 1993 [24]	autocorrelative	S	tempo, measure period
Parncutt 1994 [134]	rule-based	S	measure, beat
Large 1995 [107]	oscillator	S	beat
McAuley 1995 [126]	oscillator	S	beat
Scheirer 1998 [157]	oscillator	A	beat
Toiviainen 1998 [169]	oscillator	S	beat
Goto 1999 [64]	multiple-agent	S	beat, half-note, measure
Eck 2000 [45]	rule-based	S	tempo
Raphael 2001 [150]	probabilistic	S+A	transcription
Seppänen 2001 [160]	histogramming	A	tatum+beat
Wang & Vilermo 2001 [177]	histogramming	A	beat
Goyon *et al.* 2002 [67]	histogramming	A	tatum
Cemgil & Kappen [33]	probabilistic	S	beat
Jensen and Andersen [87]	histogramming	A	beat
Laroche [108]	probabilistic	A	beat
Uhle and Herre [173]	histogramming	A	tatum period, tempo, time signature
Hainsworth & Macleod 2004 [70]	probabilistic	A	beat
Klapuri *et al.* 2006 [P5]	probabilistic	A	measure, beat, tatum
Seppänen *et al.* 2006 [P6]	autocorrelative	A	beat, tatum
Alonso *et al.* 2007 [11]	autocorrelative	A	tempo
Davies & Plumbley 2007 [39]	autocorrelative	A	beat
Dixon 2007 [42]	multiple agent	A	beat
Ellis 2007 [49]	autocorrelative	A	beat
Peeters 2007 [140]	autocorrelative	A	measure, beat, tatum
Seyerlehner *et al.* 2007 [163]	regression	A	tempo
Shiu & Kuo 2008 [165]	probabilistic	A	beat
Eronen & Klapuri 2008 [P7]	regression	A	tempo

3.1.4 Methods for music meter analysis

This section reviews previous work on music meter analysis and serves as an introduction to publications [P5], [P6], and [P7]. Tempo estima-

tion methods can be divided into two main categories according to the type of input they process. The earliest ones processed symbolic (MIDI) input or lists of onset times and durations, whereas others take acoustic signals as input. Examples of systems processing symbolic input include the ones by Rosenthal [153], Dixon [41], Brown [24] and Toiviainen and Eerola [170]. Some of the systems such as the one by Dixon ([41]) can be extended to process acoustic signals by employing an onset detector as a preprocessing step.

The best performance on realistic, acoustic music material is typically obtained with systems that have originally been designed to process acoustic signals. One approach to analyze acoustic signals is to perform discrete onset detection and then use e.g. inter onset interval (IOI) histogramming to find the most frequent periods, see e.g. [161]. However, it has been found better to measure musical accentuation in a continuous manner instead of performing discrete onset detection [68].

The broad approaches of meter analysis systems could include

- rule-based
- autocorrelative
- oscillating filters
- histogramming
- multiple agent
- probabilistic
- regression

This is the categorization proposed by Hainsworth ([69]) with the addition of the regression category. There are methods that do not nicely fit into any of these categories, but we consider the categorization to be useful anyway for characterizing some of the most prominent aspects of the systems.

Another method of classifying meter analysis systems is by causal operation [69]. If a system is causal, the meter estimate at a given time depends only on past and present data. A noncausal system can use future data and backward decoding. In some applications, such as automatic accompaniment, causal operation is essential. In others, such as producing rhythm related metadata for digital music archives, the methods can be noncausal.

Table 3.1 presents a hopefully representative set of the various approaches.

Rule-based

Rule-based approaches tend to be simple and encode sensible music-theoretic rules [69]. They were among the first approaches to meter analysis. An example of a rule-based system is the one by Parncutt who devised a model to predict the tactus and measure for a series of repeated rhythms [134]. A simpler model for tempo prediction from symbolic data was presented by Eck [45].

Autocorrelation

Autocorrelation is a method for finding periodicities in data and has been applied in many meter analysis systems [69]. The autocorrelation provides information only on the periods, therefore phase estimation requires further processing. The lag which maximizes the autocorrelation value often coincides with the beat, although there are peaks at integer multiples of the beat. Davies and Plumbley try to explicitly model the ideal outputs of an autocorrelation function to different metrical structures using comb filter templates [39]. Brown used the autocorrelation to predict the beat and measure period from single melodic lines in symbolic format [24]. Ellis first estimates the beat period using autocorrelation and then finds the individual beats using dynamic programming [49]. Alonso *et al.* use a subspace analysis method to perform harmonic+noise decomposition before accent feature extraction and periodicity analysis using autocorrelation or other related periodicity estimators [11]. Peeters proposes the combination of DFT and autocorrelation for period estimation to suppress the harmonics in the periodicity observation [140].

Oscillating filters

Two distinct approaches can be found in oscillating filter methods for meter analysis [69]. One is based on exciting an adaptive oscillator by an input signal and, if successful, the oscillator starts to resonate at the frequency of the beat. Large used a single nonlinear oscillator with adaptive phase and period to track the beat of piano performances represented in symbolic format [107]. In his method, a sequence of impulses at note onsets acted as a driver and perturbed both the period and phase of an oscillator. Other examples of methods using an adaptive oscillator include those by McAuley [126] and Toiviainen [169]. The input to the systems by Large ([107]) and McAuley ([126]) consisted of series of impulses each corresponding to an onset of an individual note. The goal of Toiviainen was to build an interactive MIDI accompanist that tracks the performance in real time and plays back a predefined accompaniment in synchrony with the performance [169].

The second approach of oscillating filters is based on using a bank of comb filter resonators with delays spanning the range of periods to be estimated. This approach was pioneered by Scheirer who implemented one of the first successful methods for beat tracking from audio [157]. The output of a comb filter with delay τ for input $v(n)$ is given by

$$r(\tau, n) = \alpha_\tau r(\tau, n - \tau) + (1 - \alpha_\tau)v(n) \tag{3.2}$$

where the feedback gain $\alpha_\tau = 0.5^{\tau/T_0}$ is calculated based on a selected half-time T_0 in samples. The comb filters have an exponentially decaying impulse response and the half-time refers to the delay during which the response decays to half of its initial value. Scheirer used a half-time equivalent to 1.5–2 seconds in his beat tracking system [157]. In publication [P5] we use a half-time equivalent to 3 seconds since the goal is to analyze also longer, measure level pulses. A bank of comb filters can be used as a periodicity estimator when the delays τ are set so that they get values across the range of possible periods to be estimated. The comb filter which gives the most energetic output is likely to correspond to the beat period or its multiple or sub-division. Moreover, an estimate of the phase is available by examining the internal state of the delay of the most energetic comb filter [157]. This method is well suited for causal beat tracking. A disadvantage of this method that it is computationally intensive, especially if the comb filter bank is used to process several frequency bands separately as proposed by [157]. McKinney and Moelants compared the tempo histograms obtained from tempo tapping data of human subjects and periodicity outputs of a comb filterbank, autocorrelation, and an IOI histogram, and concluded that the output of a comb filterbank was closest to the tempo histogram obtained from human subjects [127].

A bank of comb-filters performs well in period and phase estimation [P5], but is computationally intensive. In publication [P6] a computationally lighter solution combining autocorrelation and discrete cosine transform is used for periodicity estimation. For phase estimation, we still use comb-filters but now in an adaptive manner, tuning the comb-filter parameters according to the current and previous period estimates.

Histogramming

Histogramming methods are based on detecting discrete onsets from the input signal and histogramming the inter-onset-intervals of detected onsets to find the most prominent periodicity. A good example of this category of methods is the one by Seppänen [160]. He first performed onset detection followed by tatum period estimation by IOI histogramming. Several acoustic features were then extracted at locations defined

by the tatum signal and used in a pattern recognition system to classify which of the tatum instances corresponded to beats. Seppänen reports that the method did not match the Sheirer method in beat tracking performance. Other examples of histogramming methods include the ones by Goyon *et al.* [67], Wang and Vilermo [177], Uhle and Herre [173], and Jensen and Andersen [87].

Multiple agent

The basic idea of multiple agent methods is that there are multiple agents or hypotheses independently tracking the beat [69]. Each agent receives scores based on how well it fits to the data. Low scoring agents may be killed during the process. At the end of the signal, the agent with the highest score wins and determines the beat. Early methods operating on symbolic or MIDI input include e.g. the ones by Allen and Dannenberg [10] and Rosenthal [153]. Later methods operating on audio signals include those by Goto [64] and Dixon [42][1].

Goto first performs onset detection on several frequency ranges [64]. The onsets are then fed to multiple agents which make parallel pulse hypotheses based on the onset time vectors. The agents calculate the inter-beat interval and predict the next beat time. Information on harmonic changes is used to determine the type of the pulse (beat, half note, or measure) and estimate the hypothesis reliability [64].

Dixon has developed a method called BeatRoot which uses a multiple agent architecture [42]. The first versions of the method processed MIDI input. In the latest version a spectral-flux based onset detector is used to make the system applicable for beat tracking on audio.

Probabilistic

Probabilistic methods define a model for the meter process whose parameters are then estimated. The basic idea here is that the underlying meter process goes through a sequence of states, and generates a sequence of observations such as periodicity vectors or onset times. Cemgil and Kappen ([33]) formulated a linear dynamic system for beat tracking which has since been used by other authors such as Shiu and Kuo ([165]) and Hainsworth and Macleod ([70]). The beat process is modeled as a linear dynamic system as follows:

$$\mathbf{x}_{n+1} = \mathbf{\Phi}(n+1|n)\mathbf{x}_n + \epsilon_n, \quad \text{(transition equation)} \tag{3.3}$$

$$\mathbf{y}_n = \mathbf{M}(n)\mathbf{x}_n + \mathbf{v}_n, \quad \text{(observation equation)} \tag{3.4}$$

[1] Goto and Dixon have written many early papers on the topic: we selected here the most recent and representative articles.

where $\mathbf{x}_n$ is the hidden state variable and $\mathbf{y}_n$ the observation, and ϵ_n and $\mathbf{v}_n$ are noise terms. The state variable is

$$\mathbf{x}_n = [\phi_n, \tau_n]^T, \tag{3.5}$$

where ϕ_n and τ_n are the phase (temporal location) and period of the current beat, respectively. The next beat location is predicted as

$$\phi_{n+1} = \phi_n + \tau_n \tag{3.6}$$

and the next period as the previous period, i.e., $\tau_{n+1} = \tau_n$. Consequently, the state transition matrix $\mathbf{\Phi}(n+1|n)$ can be written as

$$\mathbf{\Phi}(n+1|n) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}. \tag{3.7}$$

Shiu and Kuo ([165]) and Hainsworth and Macleod ([70]) observe only onsets and not the period. Then,

$$\mathbf{M}(n) = \begin{bmatrix} 1 & 0 \end{bmatrix}. \tag{3.8}$$

Thus, the observation $\mathbf{y}_n$ is the nth observed onset time and corresponds to the ϕ_n in $\mathbf{x}_n$. Beat tracking according to this model consist of the sequential estimation of the state trajectory between times 0 and n. This is solved with Kalman filtering in ([165]) and with particle filtering in [70].

A different probabilistic formulation is presented in [P5]. There, the meter process is modeled as a hidden Markov model depicted in Figure 3.5. The hidden variables are the tatum, beat (tactus), and measure periods, denoted by τ^A, τ^B, and τ^C, respectively. The observation is the periodicity vector (output of the resonance filterbank) s. Arrows indicate dependencies between the variables. The transition probabilities of the model are designed to impose smoothness on the adjacent period estimates, and to model the dependencies of the different pulse levels. The optimal sequence of period estimates is found by Viterbi decoding through the model. Thus, the model estimates the periods of the three pulses simultaneously. The phase estimation is done after period estimation. Two separate hidden Markov models are evaluated in parallel, one for the beat phase and another for the measure phase. In both models, the observation consists of the bandwise output of the resonator corresponding to the found pulse period. Transition probabilities are designed to impose smoothness between successive phase estimates. Phase estimates are obtained by Viterbi decoding through the beat and measure phase models. Hainsworth ([69]) reports that the method in [P5] outperforms his method in [70]. However, since the observations fed to these two probabilistic models are different, we cannot

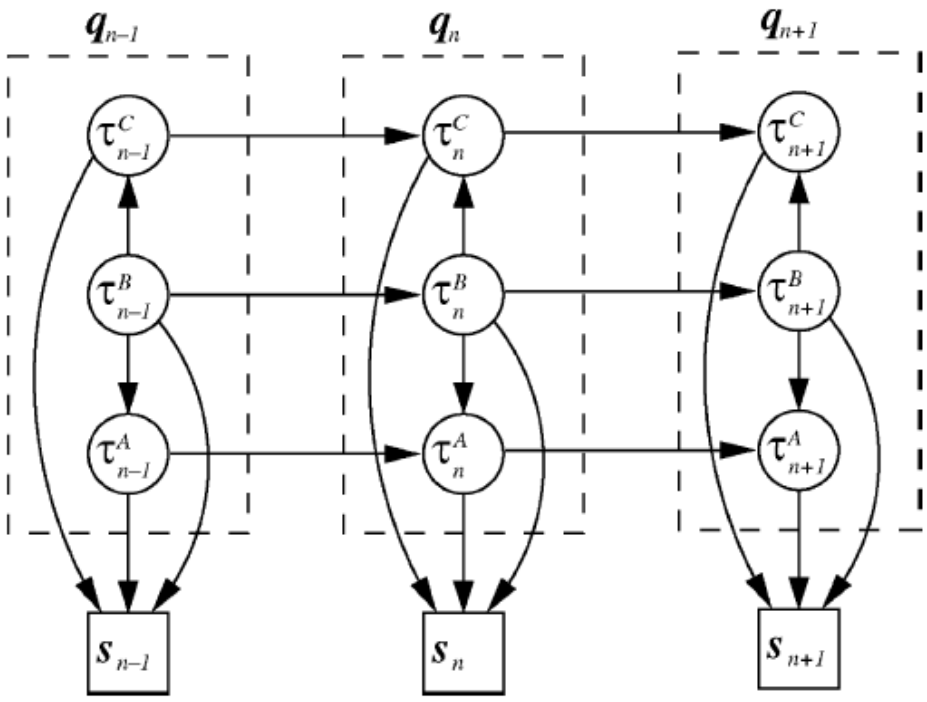


Figure 3.5: Hidden Markov model for the temporal evolution of the tatum, beat, and measure pulse periods. Reprinted from [P5]. ©2006 IEEE.

yet draw general conclusions on what is the best probabilistic model for the meter process.

Other approaches for meter analysis which could be categorized as probabilistic include those by Raphael ([150]), who performed rhythm transcription with a hidden Markov model that described the simultaneous evolution of three processes: a rhythm process, a tempo process, and an observable process. The rhythm process modeled the position within a measure a note can have. The observation was IOI data measured from MIDI or from audio with the help of an onset detector. Laroche modeled an ideal accent signal as a sequence of discrete pulses, which was then correlated with the measured accent signal to determine a set of beat period candidates. Based on the beat period candidates, dynamic programming was applied to find the beat phase [108].

Regression

We add here a new category of tempo estimators which is based on using regression. Seyerlehner *et al.* proposed the k-Nearest Neighbor algorithm as an interesting alternative to peak picking from periodic-

ity functions [163]. Peak picking stages are error prone and one of the potential performance bottlenecks in rhythm analysis systems. For example, an autocorrelation type beat tracker may select the beat period by picking the maximum peak from the autocorrelation function, possibly weighted by the beat prior. Using the k-Nearest Neighbor was motivated based on the observation that songs with close tempi have similar periodicity functions. The authors searched the nearest neighbors for a periodicity vector and predicted the tempo according to the value that appeared most often within the k songs but did not report significant performance improvement over reference methods. Publication [P7] studies this approach further and shows significant improvement in tempo estimation accuracy over the method presented in [P5].

3.2 Structure analysis and music thumbnailing

This section describes the necessary background and related research for the chorus detection method presented in publication [P8].

3.2.1 Overview

Music thumbnailing refers to the extraction of a characteristic, representative excerpt from a music file. Often the chorus or refrain is the most representative and ”catchiest” part of a song. A basic application is to use this excerpt for previewing a music track. This is very useful if the user wishes to quickly get an impression of the content of a playlist, for example, or quickly browse the songs in an unknown album. In addition, the chorus part of a song would often make a good ring tone for a mobile phone, and automatic analysis of the chorus section would thus facilitate automatical extraction of ring tone sections from music files.

Western popular music is well suited for automatic thumbnailing as it often consists of distinguishable sections, such as intro, verse, chorus, bridge, and outro. For example, the structure of a song may be intro, verse, chorus, verse, chorus, chorus. Some songs do not have as clear verse-chorus structure but there still often exist separate sections, such as section A and section B which repeat. In this case the most often repeating and energetic section is likely to contain the most recognizable part of the song.

The goal of music structure analysis is to analyze the location of one or more sections from the music file. The methods typically start by computing features from the signal using either fixed-length frames or beat-synchronous frames. Next, the goal is to find the segment boundaries and to group repeating segments, such as all choruses. Peeters *et al.* ([141]) divide the methods into two main categories: the ”state”

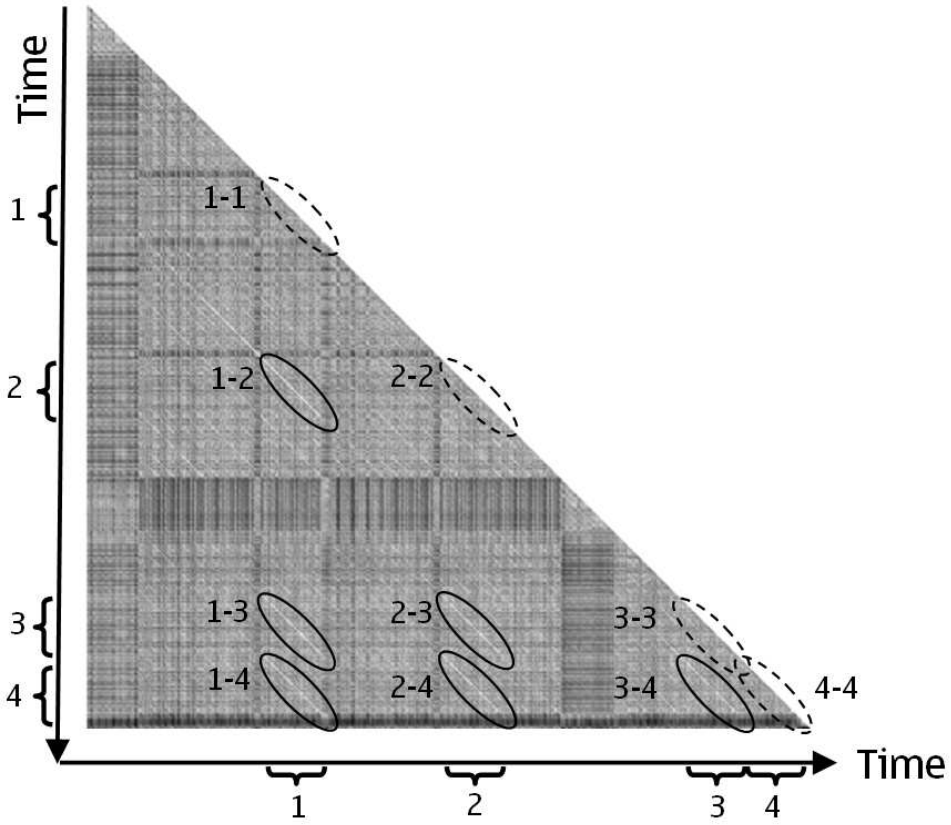


Figure 3.6: Self-distance matrix of the song ”Superstar” by Jamelia. The ellipses mark the diagonal stripes of low-distance corresponding to chorus repetitions. This particular song has four choruses marked with 1, 2, 3, and 4. The notation x-y indicates that the particular diagonal stripe is caused by a low distance between the chorus instances x and y. The dashed ellipses indicate low distance stripes caused by matching a chorus to itself which are hidden by the main diagonal.

approach and the ”sequence” approach. The state approach considers each part of a music track to be generated by a state. Each state has characteristic acoustic information which separates the parts generated by different states from each other. A part does not have to be repeated later in the track. Representing musical parts as states with different acoustic properties is motivated by the knowledge that in popular music the different parts often have a characteristic accompaniment which stays constant during the part. In this case, the goal of the structure analysis is to find the most likely state sequence that could have generated the acoustic features. A good example of the state approach is the

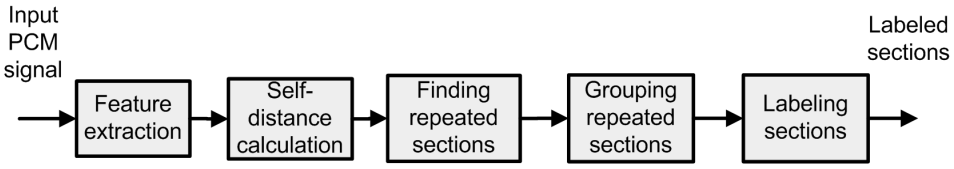


Figure 3.7: A schematic view of a music structure analyzer.

work by Logan and Chu who used agglomerative clustering or a hidden Markov model and Viterbi decoding to assign feature frames to different segments [117]. A basic problem especially with the HMM based segmentation is how to constrain the temporal span of the segmentation to be long enough. When a HMM is trained using short-time features for a music file, similar low-level feature vectors may be grouped to the same state but it is unlikely that this would match with high-level song segment structure. For example, in Figure 2.6 different states model different parts of the trumpet notes. One solution is to use a large number of states in the HMM model, and then histogram the decoded sequence of states and use the histograms as new features [112].

The sequence approach assumes that there exist repeating sequences in the music track. A sequence is defined as a time interval with certain succession of musical properties, such as notes or chords. Different repeats of a sequence are not necessarily identical but similar. These sequences are visible in a self-distance matrix (SDM) as off-diagonal stripes indicating a succession of pairs of times with high similarity. Figure 3.6 shows an example SDM for the song "Superstar" by Jamelia.

We will give here a short introduction to the steps of a music structure analysis method which is based on the sequence approach and SDM processing. This serves as an introduction to [P8]. Figure 3.7 depicts the basic operations of a music structure analyzer that is based on self-distance analysis [65, 136]. The method starts with feature extraction and SDM calculation. This is followed by finding repeated sections from the SDM, grouping repeated sections belonging to the same high-level segment (e.g. verse), and selecting the chorus sections. The following sections describe these steps in more detail.

3.2.2 Chroma feature extraction

Whereas MFCCs are applicable to a wide variety of sounds such as speech, music, and environmental sounds, chroma is a music-specific feature for describing the spectral content of musical sounds. The chroma is motivated by the Shephard helix ([164]) of musical pitch perception [66]. Chroma features are typically used for music structure analysis ([16,

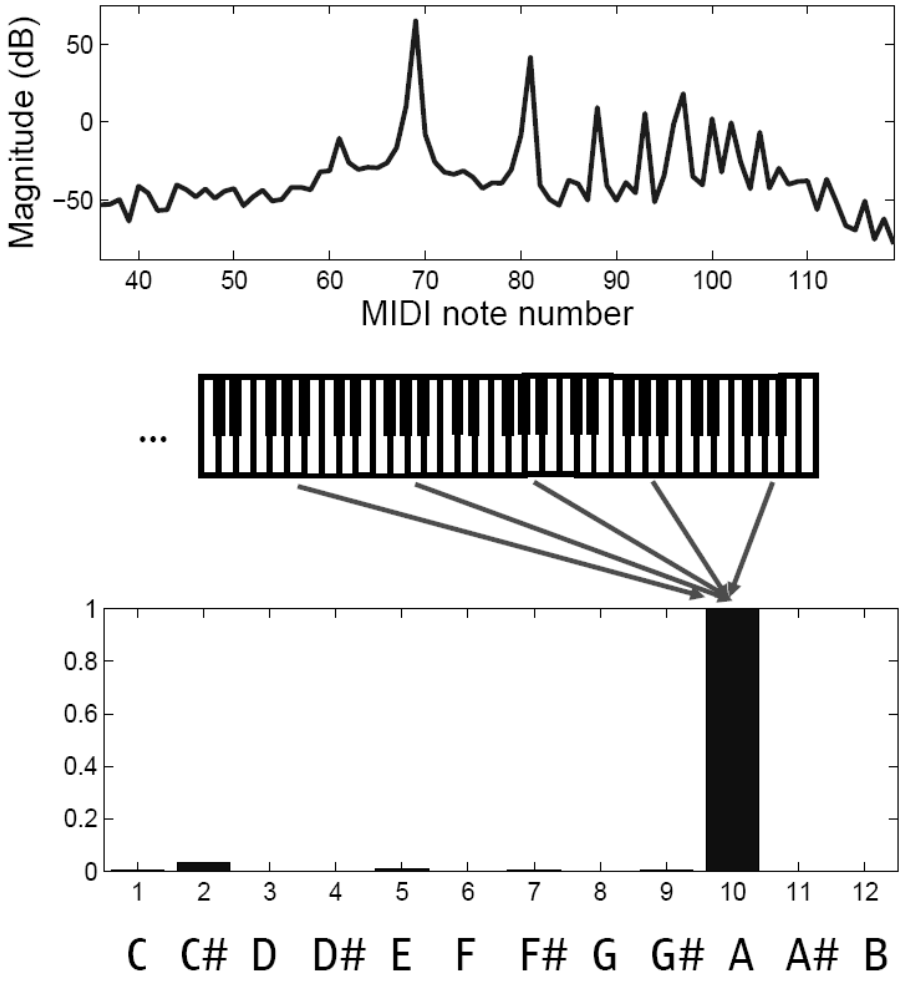


Figure 3.8: A schematic view of chroma feature analysis. The top panel shows the magnitude spectrum of a note A4 with fundamental frequency 440 Hz. The energy corresponding to the same pitch class is accumulated over several octaves on the same pitch class bin.

66]), key estimation (e.g. [62, 138]), cover song identification ([84]), or detecting harmonic changes for bar line analysis [83]. Figure 3.8 depicts a schematic view of the chroma feature analysis. Energy at a musical semitone scale is accumulated to twelve pitch classes over a range of octaves [16]. In the figure, the note frequency is represented as MIDI note number. The conversion from frequency in Hertz to MIDI note number is done using the equation

$$\text{MIDI note number} = 69 + round(12\log(\frac{f}{440})/\log(2)), \qquad (3.9)$$

where $round$ denotes rounding to nearest integer. Note that using MIDI note numbers is not necessary for chroma feature analysis but is used in the figure for the convenience of representing the x-axis.

A straightforward way of calculating the chroma features is to map each bin of a short-time discrete Fourier transform to exactly one of the twelve pitch classes C, C#, D, D#, E, F, F#, G, G#, A, A#, B, with no overlap. A relatively long analysis frame is needed to get sufficient resolution for the lower notes. In [P8] we use 186 ms frames. The energy is calculated from a range of six octaves from C3 to B8 and summed to the corresponding pitch classes. The chroma vectors are often normalized by dividing each vector by its maximum value.

Another alternative for calculating the the chroma features is to use a multiple fundamental frequency estimator to estimate the strength of a range of F0 candidates, which are then folded to chroma bins. This kind of approach was proposed by Paulus and Klapuri for music structure analysis in [136]. We apply their chroma analysis method as a first step in musical accent feature estimation in [P7]. The input signal sampled at 44.1 kHz sampling rate and 16-bit resolution is first divided into 93 ms frames with 50% overlap. In each frame, the salience, or strength, of each F0 candidate is calculated as a weighted sum of the amplitudes of its harmonic partials in a spectrally whitened signal frame [98]. The range of fundamental frequencies used here is from 80 Hz to 640 Hz. Next, a transform is made into a musical frequency scale having a resolution of 1/3rd of a semitone (36 bins per octave). For each bin, only the maximum-salience fundamental frequency component is retained. Finally the octave equivalence classes are summed over the whole pitch range using a resolution of three bins per semitone to produce a 36 dimensional chroma vector $x_b(k)$, where k is the frame index and $b = 1, 2, ..., b_0$ is the pitch class index, with $b_0 = 36$.

There exist several variants for measuring information similar to the chroma feature. The pitch class profile (PCP) is a synonym for the chroma features [59]. Gomez calls her variant of the chroma feature analysis the harmonic pitch class profile (HPCP) [63]. Purwins *et al.* compute a twelve-dimensional chroma representation from the constant Q transform ([23]) and call the features constant-Q profiles [147]. The constant-Q transform achieves a constant-Q resolution whereby time resolution increases and frequency resolution decreases with increasing frequency (Q denotes the ratio of frequency to resolution). Moreover, the frequencies of the transform bins can be made to coincide with musical frequencies. The pitch histogram by Tzanetakis and Cook, which is based on detecting and histogramming dominant pitches from the output of a multiple F0 estimator, is also a closely related feature [172].

In music structure analysis, it is desired that the distance would be high between different song segments (e.g. verse and chorus) and

small between instances of the same segment (e.g. different repetitions of the chorus). The chroma features reveal similarities in melody and harmonic accompaniment between different sections of the song even if the used instrumentation or lyrics would change between sections. The MFCC features are sensitive for changing accompaniment between different choruses and differences in lyrics at different instances of the verse. Bartsch and Wakefield reported that chroma features outperform MFCC features in music thumbnailing [16]. Most current structure analysis methods use chroma features and optionally augment them with MFCC features or features describing the rhythm. Paulus and Klapuri have presented a detailed study of the suitability of different features for music structure analysis [135].

3.2.3 Self-similarity analysis

The next step is to calculate song self-similarity (or equivalently self-distance). Various distance functions such as the Euclidean distance or the cosine distance (inner product) can be used. Specialized distance functions have been presented by Goto ([66]) and Lu *et al.* [118]. Before distance calculation, the feature vectors are usually normalized, e.g., to a mean of zero and standard deviation of one, or to a maximum element of one.

The self-distance measurements can be represented in a self-distance matrix (SDM). Figure 3.6 shows an example SDM for the song "Superstar" by Jamelia. Each entry $D(i, j)$ in the SDM represents the distance of the beat synchronous features of two time instances i and j of the music file. See details in publication [P8]. The song has four choruses, which repeat with almost the same melodic, harmonic, and instrumentation content, resulting in strong diagonal segments of low self-distance into the SDM. A diagonal segment which starts at the point (i, j) and ends at $(\acute{i}, \acute{j})$ indicates that the musical segment which starts at time i and ends at $\acute{i}$ repeats starting at time j and ending at time $\acute{j}$. The diagonal stripe is created to the SDM since the feature vector sequences during these time intervals are similar.

There are also diagonal segments of low self similarity corresponding to the verse, see e.g. the diagonal stripe before the stripe corresponding to the repetition of choruses 1 and 2. In addition, there are usually many short segments of low self-distance corresponding to repeated melodic and/or rhythmic phrasing.

If a song has varying tempo and constant-length analysis frames are used, the diagonal stripes in the self-distance matrix will not be diagonal anymore but curved according to the tempo changes. Beat synchronous feature segmentation will keep the stripes diagonal.

Bartsch and Wakefield ([16]) and Goto ([66]) used a representation

equivalent to the SDM called time-lag triangle. In the SDM both axes represent time, in the time-lag triangle the axes are time and lag. The matrix $D(i, j)$ can be converted to a time-lag triangle $L(l_{ij}, j)$ with $l_{ij} = i - j$. The time-lag triangle transforms a diagonal repetition into a horizontal constant-lag line.

3.2.4 Detecting repeating sections

The next step is to detect repeating segments from the SDM or time-lag triangle. This is not a straightforward task since the diagonal stripes corresponding to repetitions can be very weak when the features are extracted from realistic audio recordings. A straightforward method would be to binarize[2] the SDM using some known methods for image binarization. However, the problem is that this will create many erroneous detected regions of small self-distance in locations where just a few feature vectors happen to be similar to each other. A better alternative is to utilize the knowledge that we are looking for diagonal stripes of low self-distance. Bartsch and Wakefield proposed to calculate moving averages of the SDM values [16]. Goto proposed a two-stage adaptive thresholding method where sums are calculated across the diagonals of the SDM, and adaptive thresholding is then applied to detect a certain number of diagonals to be searched for repetitions [66]. The final repetitions are searched using another adaptive thresholding on the selected one-dimensional diagonal segments of the SDM. A slightly varied version of this method is used in [P8]. Note that both Bartsch and Wakefield and Goto proposed the methods for the time-lag triangle, but this is an equivalent representation to the SDM [65].

3.2.5 Grouping and labeling sections

Each diagonal segment in the SDM represents just a pair of repeated sections. If a complete description for the musical pieces is desired, next it is necessary to group the segments representing the same musical section. Cooper and Foote construct a segment level distance matrix and apply the Singular Value Decomposition to cluster similar segments [36]. Goto groups together segments detected from the time-lag triangle having close starting and ending points, and in addition utilizes knowledge of already found segments to search for missing segments [66].

The remaining problem is how to assign meaningful labels to the sections. Most studies have considered only labeling the chorus, and

[2]Binarization is an image processing operation during which an image consisting of multiple shades of gray is converted to one having only two levels, black and white.

used various heuristics, such as selecting the most often repeating section as a chorus [66]. Ong presents an extension of the Goto method to obtain a more complete segmentation [131]. Only few studies have attempted full description including segmentation and assigning musically meaningful labels for the segments. Examples include Maddage *et al.* who perform explicit segmentation into vocal & nonvocal sections to aid structure analysis [121], and Paulus and Klapuri who proposed using N-grams for automatic segment labeling [136].

3.2.6 Methods for music structure analysis

Table 3.2 summarizes selected research on music structure analysis and chorus detection. This is not a complete listing but hopefully a representative set of the various approaches. The methods are categorized according to the features used and the main approach, ”sequence” or ”state”. In addition, the table lists the information produced by the system. ”Thumbnail” means that the system produces a single representative section to be used as a thumbnail. Methods that produce segmentation information return the boundaries of all musical parts, but without musically meaningful labels such as intro, verse, chorus, bridge, or outro. Some methods produce the boundaries of all parts but label only e.g. the chorus, or the chorus and verse. The methods that produce a complete description including segment boundaries and musical labels for the parts can be considered the most advanced.

One of the first examples music thumbnail extraction using clustering was that of Logan and Chu [117]. Levy *et al.* propose a hierarchical timbre model where a large HMM modeling different ”timbre types” of music is first trained, and the most likely sequence of states is obtained for a music file by Viterbi decoding through this model [112, 111]. Histograms of the decoded sequence of states are then used to characterize different musical sections. Rhodes *et al.* also use state occupancy histograms as features and propose an explicit prior probability distribution for the section durations in a Bayesian structure analysis framework [152].

Cooper & Foote [36] first calculate a self-distance matrix. Segment boundaries are found by correlating a kernel along the diagonal of the matrix. A segment-level SDM is them computed, and segments are clustered by applying singular value decomposition (SVD) on the segment-level SDM. Paulus and Klapuri present a related method where initial segments are first found by kernel correlation [136]. They define the probability that two segments belong to the same musical part as a function of distances between segments, and then try to optimize a probabilistic fitness measure for different segment description candidates using these probabilities. Jensen solves the problem using the

shortest path algorithm for the directed acyclic graph [86].

The use of the self-similarity representation for music structure analysis was proposed by Foote, who first considered using the matrix for visualizing music and audio content [54]. Several methods have been proposed to automatically extract structural information from the self-distance matrix. Wellhausen & Crysandt used the MPEG-7 spectral envelope features to calculate a similarity matrix and detected diagonal line segments from it [178]. Chai used chroma features and proposed distance function to overcome variations in the key between different occurrences of the same part [34]. Bartsch & Wakefield [16] and Goto [66] operated on an equivalent time-lag triangle representation.

Some methods apply classification of music segments to help labeling. Lu *et al.* ([118]) and Maddage ([121]) use classification between instrumental / vocal sections as further cues for segment labeling. For example, Lu *et al.* classify a segment as intro, bridge, or outro if it is classified as instrumental and depending on the temporal position.

Table 3.2: Summary of selected research on music structure analysis and chorus detection.

Author year ref.	Features	Approach	Output
Foote 1999 [54]	MFCC	sequence	visualization
Logan & Chu 2000 [117]	MFCC	state	thumbnail
Dannenberg & Hu 2002 [38]	chroma or transcription	sequence	segmentation
Peeters *et al.* 2002 [141]	bandwise FFT	state	segmentation
Cooper & Foote 2003 [36]	MFCC	state	segmentation+ verse+chorus
Wellhausen & Crysandt 2003 [178]	MPEG-7 spect. env.	sequence	choruses
Lu *et al.* 2004 [118]	CQT	sequence	segmentation+ intro+bridge+ outro
Bartsch & Wakefield 2005 [16]	chroma	sequence	thumbnail
Chai 2005 [34]	chroma	sequence	segmentation
Goto 2006 [66]	chroma	sequence	choruses
Maddage 2006 [121]	chroma+octave-scale cepstral coefficients	state	full description
Eronen 2007 [P8]	MFCC+chroma	sequence	thumbnail
Jensen 2007 [86]	rhythmogram+ PLP+ chroma	state	segmentation
Ong 2007 [131]	HPCP	sequence	segmentation
Peeters 2007 [139]	MFCC+ spect. contrast+ chroma	sequence	segmentation
Levy & Sandler 2008 [111]	MPEG-7 AudioSpec.Proj.	state	segmentation
Paulus & Klapuri 2008 [136]	MFCC+chroma+ rhythmogram	state	full description

Chapter 4

Applications

This Chapter discusses some applications of music content analysis, focusing on applications that can utilize the methods presented in this thesis. We conclude with a brief discussion on where the analysis algorithms should be run in a practical environment where the user has computing devices connected to an online music service.

4.1 Music recommendation and search

Most of the commercial interest in the music information retrieval field is probably targeted towards the problems of music recommendation and automatic playlist generation. Here, the task can be defined for example as follows: given an example song, return a list of songs with similar characteristics. A question raises how well do methods based on audio content only perform in returning similar songs. Finding similar music based on content attributes has received plenty of research interest, see e.g. [132]. A certain level of performance can be obtained using audio information for music recommendation. However, there seems to exist a ”glass ceiling” above which it is difficult to get using only low-level signal features. The performance seems to saturate around 60%-70% of good matches [15].

Similar conclusions were obtained in a user study reported by Lehtiniemi in [109]. For that study, the author of this thesis implemented a content-based music recommendation method which utilized a similarity metric proposed by Pampalk [132]. Mel-frequency cepstral coefficients are extracted from music files and modeled with single-Gaussian densities with full covariance matrices. Song distance is calculated with the Kullback-Leibler (KL) divergence between the Gaussian distributions. In addition, the rhythmic aspects of signals are modeled and compared with the so-called fluctuation pattern feature which measures the strength of amplitude modulation on a set of frequency bands. The final

distance between music files is a weighted sum of the timbral distance returned by the KL divergence and the distance of the fluctuation pattern features. To make the system scale to large music collections, a clustering scheme was implemented where the distance between songs is computed only within songs in the same cluster. The author implemented the method into a prototype end-to-end mobile music service. The users of the service were able to select a seed song and request playlists of similar music to their mobile phone and stream the songs over a network connection. The users were requested to vote whether the returned song was similar to the seed song. On the average, 63% of the songs were considered acceptable. The most annoying errors are cross-genre confusions the system makes: e.g. some classical and jazz songs are confused by the system, sometimes also music from the rap and rock genres. Within some music types such as metal, the recommendations based on content attributes only can be surprisingly good. The general conclusion is that the content attributes need to be augmented with higher level metadata such as genre and release year to make the recommendations acceptable. This kind of song similarity is not expected to suffice as the only source of music recommendations, but can be used to e.g. provide recommendations to new content for which there are not yet enough material to train a collaborative-filtering ([73]) based music recommender.

4.2 Active music listening

Active music listening can be defined as a form of music enjoyment where the listener has some control over the content besides basic transport controls of play, stop, rewind, forward, and changing the song. For example, in seamless playback or beat mixing the user may make transitions from one song to another while the system takes care of mixing the song in a continuous fashion. Beat and possibly measure level analysis is utilized to time-synchronize the beats, and time stretching (or pitch shifting) to align the tempos during the transition. In clubs and discos professional DJs vary music tracks also by looping and rearranging them. However, DJ devices are expensive and have complicated user interfaces making them unattractive for the public. The devices may require manual preparation by adding loop points or segmenting the music in advance. Some computer applications may offer a semi-automatic approach consisting of automatic beat tracking followed by a step where the user taps in the downbeats (for example, in Magix Music Maker 10). The availability of fully automatic methods for extracting music rhythm information such as beats and measures from musical files can bring these functionalities to amateur listeners.

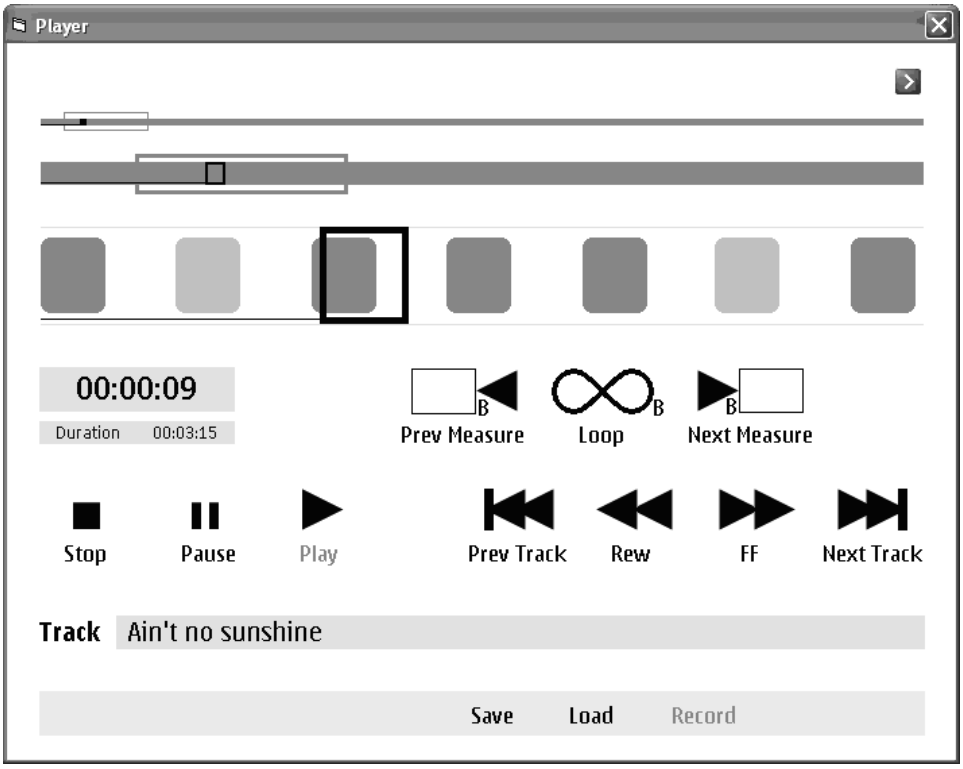


Figure 4.1: A prototype music player interface with buttons for looping and skipping measures in a beat synchronous manner.

One novel example of active music listening is presented here. The ideas were originally presented by Timo Kosonen and the author in [102]. The idea in this music player concept is to allow the user to repeat parts of the music file in an easy manner. The user interface is depicted in Figure 4.1. A loop button has been included in addition to the traditional music player controller buttons. When the user presses the loop button, the system starts to loop the currently playing measure of the music. The end result is an entertaining music listening experience especially with electronic music, where the user may easily repeat parts of the file in order to e.g. make the music file longer. For example, when a user wishes to entertain his guests in a home party, he may make simple DJ-like effects such as looping portions of music in an easy way.

Another technique examined in this prototype was to study whether the listening experience during fast forward and rewind can be made more pleasing by utilizing rhythm information. When the user enables fast forward or rewind, the system will proceed as follows: it will first render the currently playing beat until the end, and then skip to the beginning of the next measure in the case of fast forward, or in the

beginning of the previous measure if the user initiated rewind, and play the first beat. Then it jumps again to another measure, plays the first beat, and so on. The audible effect of this compared to the conventional method of fast forward or rewind is that the user is able to hear the tempo of the piece during fast forward or rewind.

Tzanetakis and Cook ([171]), Goto ([66]), and Boutard *et al.* ([22]) have studied a skip to section functionality for efficient music browsing. Displaying musical sections with different colors and allowing the user to skip between the sections help the user find a section of interest within a music track. Wood and O'Keefe extended an open source music player with a "mood bar" that presents a graphical mapping of a low-level feature along the music timeline [180]. However, their publications do not discuss using musical meter for intra-track skipping. Moreover, their implementations do not keep the music playback continuous when the user presses a skip button; their implementations may be good choices, if the goal is only to allow the user to locate a section of interest quickly. The focus of the presented active music listening interface is more on entertainment; we want to make intra-track music browsing more pleasing by preserving the rhythm sensation, and allow the user to focus on a particular section by a looping functionality.

4.3 Music variations and ring tone extraction

Jehan has presented methods to manipulate music recordings, for example, by creating "music textures" that continue infinitely by concatenating music track segments, pieces of music tracks between onsets, with a similar metrical location [83]. For example, a music track segment occurring on a downbeat is a candidate for occurring on a downbeat in the extended music texture. The presented music player interface did not utilize segmentation into the sound onset level but created longer versions of music tracks by repeating full measures or varied versions of music tracks by changing the playback order of measures or full beats.

The bottom part of the user interface in Figure 4.1 has simple controls for recording the playback order of the song. During or before playback, the user may press record and the system records which musical segments (beats and measures) are rendered and in which order. This allows the user to record a personalized version of a song by looping some segments, or to record only a part of the song to be used as a personalized ring tone. The idea in the player interface is that the variation is not stored as audio file but as metadata: a simple metadata format indexing the beats, measures, and sections of the songs can be used to store the variations in a compact way.

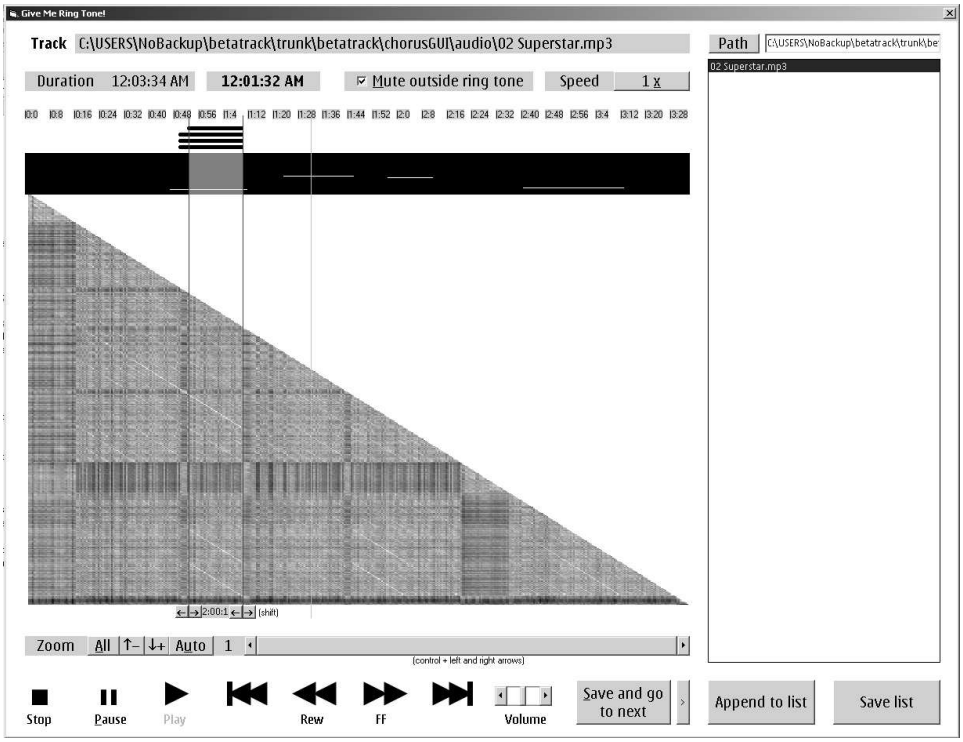


Figure 4.2: A prototype tool for visualizing, listening, and fixing automatically analyzed chorus segments for ring tone use.

One of the interesting uses for chorus detection is automatic extraction of ring tone segments from arbitrary music files. The chorus is often the most catchiest and memorable part of a song and thus suitable to be used as a ring tone. However, as the analysis methods are not perfect and especially determining the accurate boundaries of the chorus start and end section is challenging (see details in [P8]), there may be a need for tools to perform further adjustments to the analyzed chorus section. Figure 4.2 presents a prototype user interface developed by Timo Kosonen and the author which can be used to visualize, listen to, and correct chorus section analysis results. The motivation of this interface was twofold: to operate as a chorus annotation tool to provide evaluation material for the algorithm, and to test the feasibility of a semiautomatic ring tone creation scheme where an algorithm is run first and then a manual inspection is done to verify the result.

The user interface provides mechanisms to make checking the suitability of the chorus section as a ring tone very fast. With the press of the space bar, the operator can start playing the chorus section from the beginning. Special buttons exist to adjust the beginning and end of the chorus section backward and forward. Moreover, when moving the

location of the chorus section, the system automatically quantizes the start to the nearest beat. This makes it possible to very fast adjust the location so that the sample beginning remains continuous and does not cause clicks: the playback can often be started in the beginning of a full beat in a smooth manner.

4.4 A note on practical implementations

Considering practical mobile music services and applications that use automatically analyzed music metadata, there are several alternatives on where the analysis algorithms should be run. It is possible to run analysis the on the mobile device for the user's music files. The benefit of this is that it does not matter where the material has come from, the same analysis can be performed for over-the-air (OTA) downloaded content or content transferred from a PC, or even content recorded with the mobile computer. However, the analysis consumes battery power, more complex algorithms are slow to run on current devices, and, if the content is protected with some digital rights management (DRM) technology, the audio waveform may not be accessible.

There are certain applications where it is convenient to perform the analysis on the mobile computer. One example are various beat synchronized visualizations where changes in the graphics are synchronized to the beat of the music. In this case the beat tracking can be performed in real time during the music playback and rendering of the visualization. For beat tracking this is feasible as it can be performed computationally efficiently while maintaining sufficient analysis accuracy, as is demonstrated in publication [P6].

Another alternative is to run analyzers on the user's personal computer (PC). There we have more computing power than on the mobile terminal. However, the disadvantage is that songs downloaded OTA to the mobile device would have to be separately transferred to the PC for analysis, and then the analysis results synchronized back to the mobile device. In addition, this requires that the user has an additional device in addition to the mobile phone to be able to use those features of the application that require the metadata.

A good place to run the analysis is on the servers of a music service or a specific metadata provider. This has the benefit of having to run the analysis only once on each music file, and the analysis results can be delivered to each user needing the file metadata. The metadata can be downloaded specifically to the mobile computer or PC, or it can be attached to the header section when downloading the files. On a service, we can utilize parallelism and huge computer farms to run even complex analyzes to large catalogues. In addition, since many music

feature extractors are still in their very early stages and provide information that is semantically on a low level, there is the possibility to create part of the metadata manually by employing human experts. The process could even be a semiautomatic one where an automatic analyzer is run first, and then a human operator checks the results. One of the purposes of the tool presented in figure 4.2 was to test this kind of a semiautomatic process. The problem is, however, that the cost will be at a significantly higher level compared to fully automatic processes if we need to include a step where a human operator is needed.

A challenge when providing metadata from a service is that one needs a reliable mechanism to identify user's own music files such that metadata can be downloaded for those. Audio fingerprinting ([30][29]) is the only reliable solution, but this again consumes battery power. A problem occurs with non-commercial content such as user-created mixes or amateur production that are not found in the catalogue. A solution to this would be to send the content from the mobile terminal to the service for analysis but this consumes the scarce upstream network bandwidth. However, we have already seen the first services that provide analyses for user's own files, see The Analyze API by the Echonest corporation [7].

Chapter 5

Conlusions and future work

5.1 Conclusions

This thesis presented several methods for audio classification and music content analysis. As suggested by human timbre perception experiments, utilizing both spectral and temporal information is beneficial in musical instrument classification. A wide set of features was proposed and implemented in publication [P1] resulting in very good performance on the McGill University Master samples collection. Furthermore, experiments were carried out to investigate the potential advantage of a hierarchically structured classifier, from which we could not obtain benefits in terms of classification performance.

In [P2], we studied the importance of different features for musical instrument recognition in detail. Warped linear prediction based features proved to be successful in the automatic recognition of musical instrument solo tones, and resulted in better accuracy than what was obtained with corresponding conventional LP based features. The mel-frequency cepstral coefficients gave the best accuracy in instrument family classification, and would be the selection also for the sake of computational complexity. The best overall accuracy was obtained by augmenting the mel-cepstral coefficients with features describing the type of excitation, brightness, modulations, synchronity and fundamental frequency of tones. However, a problem remains on how to generalize across instruments and recording locations: as more than one example of an instrument are included in the evaluation the performance of the system significantly drops. This effect is evident from the performance evaluations in [P2] where the overall accuracy is significantly lower than in [P1].

The use of left-right hidden Markov models for instrument note mod-

eling was proposed in publication [P3]. In addition, we studied two computationally attractive methods to improve the performance of the system: using linear transforms to transform catenated MFCC and ΔMFCC coefficients and discriminative training of the HMMs. Transforming the features to a base with maximal statistical independence using independent component analysis can give an improvement of 9 percentage points in recognition accuracy in musical instrument classification. Discriminative training of HMMs can improve the performance when using models with a small number of states and component densities.

The audio classification system proposed in [P3] is generic and was applied to audio-based context recognition in [P4]. Contrary to musical instrument sounds, no clear benefit is obtained by using linear feature transforms when classifying environmental sounds. Discriminative training can be used to improve the accuracy when using very low-order HMMs as context models, which may be necessary on resource constrained mobile devices.

The general conclusion from [P4] is that building context aware applications using audio is feasible, especially when high-level contexts are concerned. In comparison with the human ability, the proposed system performs rather well (58% versus 69% for contexts and 82% versus 88% for high-level classes for the system and humans, respectively). Both the system and humans tend to make similar confusions mainly within the high-level categories. The recognition rate as a function of the test sequence length appears to converge only after about 30 to 60s. This poses a challenge for automatic systems since we would like to minimize the amount of time the feature extractor is running to save the battery power.

Publications [P5] to [P7] present several methods for music meter analysis. Publication [P5] presents a complete meter analysis system which performs the analysis jointly at three different time scales. The probabilistic model represents primitive musical knowledge and is capable of performing joint estimation of the tatum, tactus, and measure pulses. Several assumptions and approximations were presented to obtain reasonable model parameters with limited amount of training data. The system won the ISMIR 2004 and MIREX 2006 tempo induction contests.

Publication [P6] presented a computationally efficient method for beat and tatum estimation. A simplified back-end for beat and tatum tracking was presented. The computationally intensive bank of comb-filter resonators was substituted with a discrete cosine transform periodicity analysis and adaptive comb filtering. The back-end incorporated similar primitive musicological knowledge as the method presented in cite [P5], but with significantly less computational load. A novel method

based on adaptive comb-filtering was presented for beat phase estimation. Complexity evaluation showed that the computational cost was less than 1% of two reference methods. A real-time implementation of the method for the S60 smartphone platform was written.

The regression approach for tempo estimation proposed in [P7] was found to be superior compared to peak picking techniques applied on the periodicity vectors as is done e.g. in [P5] and [P7]. We conclude that most of the improvement is attributed to the regression based tempo estimator with a smaller contribution to the proposed F0-salience chroma accent features and GACF periodicity estimation, as there is no statistically significant difference in error rate when the accent features used in [P5] are combined with the proposed tempo estimator. In addition, the proposed regression approach is straightforward to implement and requires no explicit prior distribution for the tempo as the prior is implicitly included in the distribution of the k-NN training data vectors. The accuracy degrades gracefully when the size of the training data is reduced.

In publication [P8] we presented a computationally efficient and robust method for chorus section detection. The method analyzed song self distance by summing the self-distance matrices based on the MFCC and chroma features. A scoring method for selecting the chorus section from several candidates was proposed. In addition, a method utilizing a matched filter for refining the location of the final chorus section was proposed. The method provides accuracies sufficient for practical applications while being fast to compute.

A motivation for our research has been to study which music descriptors can be estimated robustly enough for practical applications. Tempo and chorus section estimation accuracies reach a level of 80% or beyond which starts to be sufficient for practical applications, such as active listening or music search. Music tempo perception is ambiguous also for human subjects which makes it possible that we are approaching the practical limits of obtainable performance. The chorus detector is applicable to music preview and thumbnailing for popular and rock music especially if a fade-in and fade-out is applied at the chorus boundaries. For automatic ring tone segment analysis there are more strict requirements for the beginning and end of the segment, and the performance is not yet sufficient. A semiautomatic annotation interface was presented in section 4.3 as one possible solution.

In many methods special emphasis was put on keeping the methods computationally efficient. In section 4.4 we discussed the benefits and disadvantages of alternative locations for running automatic music content analyzers: these are a mobile device, a PC computer, or a dedicated centralized server. Irrespective of where the analyzers are run, computational efficiency is important. On a mobile device it is vital in

order to keep the battery consumption low, on a PC computer an important part of the user experience is created by the application responding fast, and on a server we need to analyze catalogues of several million files. Publications [P3] and [P4] proposed the linear feature transforms and discriminative training of HMMs as potential sources for improvement in non-speech audio classification tasks with negligible additional computational cost in the on-line classification stage. Publication [P6] demonstrated how the performance of a beat tracking system can be kept at a good level while making a drastic reduction in computational cost. Publication [P8] presents a method for chorus detection which performed well and runs fast enough for processing catalogues of music of the size of several million tracks.

5.2 Future work

The music content analysis methods presented in this thesis, as most other methods developed to date, operate only on the audio signal. We expect that subdomain specific music content descriptors, e.g. special methods for jazz, classical, pop and rock genres may be necessary to further boost the performance to a level needed by practical applications. On a general level, we should study ways to leverage existing textual metadata such as genre, style, or textual information from e.g. record reviews to obtain more robust analysis of music content. In addition, automatic synchronization of MIDI files to corresponding audio files may be an interesting approach to e.g. perform tempo analysis for classical music.

Context-awareness using audio is a challenging topic but automatic systems can approach the human ability as was demonstrated in [P4]. Future research will need to answer the question on whether audio-based context sensing is useful in more general use cases and application scenarios, implementing the methods in power-efficient ways on mobile devices, and combining the various sensory information in an optimal manner. In addition, we need to solve the problems related to frictional noises when the device is being carried in bags and purses. So far the research has been done on clean recordings, next we need to analyze the performance on audio collected in realistic usage scenarios.

In musical instrument recognition, the challenge is to perform reliable recognition or instrument labeling in polyphonic mixtures. We believe that one of the most potential research directions is using partially labeled data as suggested by [114]. This stems mainly from the practical difficulty of obtaining fully segmented and labeled training material. An interesting approach would be to test this approach on really large databases where the presence of a certain instrument is indicated in the

title of the track or album, or collect this information as user tags from a music service. In addition, considering practical applications being able to label the most dominant instrument may be sufficient, without having to identify all the instruments in a mixture. This would facilitate finding music with piano, or music with blues guitar and so on.

In music meter analysis, algorithm accuracy in tempo estimation starts to be sufficient for practical applications. Remaining main challenges are in beat phase estimation, and especially measure phase estimation. Estimation of the phase is important in applications where something needs to be synchronized to the tempo. One approach to improve phase estimation is the utilization of harmonic information in an efficient manner. The regression approach proposed in [P7] might be applicable to phase estimation as well.

Bibliography

[1] Allmusic. http://www.allmusic.com/.

[2] Cumulative ISMIR proceedings. http://www.ismir.net/proceedings/.

[3] Frequently asked questions at pandora.com. http://blog.pandora.com/faq/.

[4] Last.fm. http://www.last.fm/.

[5] Pandora.com. http://www.pandora.com/.

[6] Smart valley software. http://www.happywakeup.com.

[7] The Analyze API by the Echo Nest corporation. http://the.echonest.com/analyze/.

[8] K. A.G and T. Sreenivas. Music instrument recognition: From isolated notes to solo phrases. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, Montreal, Quebec, Canada, May 2004.

[9] G. Agostini, M. Longari, and E. Pollastri. Musical instrument timbres classification with spectral features. *EURASIP J. App. Signal Proc.*, (1):5–14, 2003.

[10] P. E. Allen and R. B. Dannenberg. Tracking musical beats in real time. In *Proc. Int. Comp. Music Conf. (ICMC)*, pages 140–143, Glasgow, Scotland, 1990.

[11] M. Alonso, G. Richard, and B. David. Accurate tempo estimation based on harmonic+noise decomposition. *EURASIP J. Adv. in Signal Proc.*, 2007.

[12] C. Atkeson, A. Moore, and S. Schaal. Locally weighted learning. *AI Review*, 11:11–73, Apr. 1997.

[13] J.-J. Aucouturier and B. Defreville. More on bag-of-frames: Differences in the human processing of music and soundscapes. *J. Acoust. Soc. Am.*

[14] J.-J. Aucouturier, B. Defreville, and F. Pachet. The bag-of-frame approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music. *J. Acoust. Soc. Am.*, 122(2):881–891, 2007.

[15] J.-J. Aucouturier and F. Pachet. Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences*, 1(1), 2004.

[16] M. A. Bartsch and G. H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Trans. Multimedia*, 7(1):96–104, Feb. 2005.

[17] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler. A tutorial on onset detection in music signals. *IEEE Trans. Speech and Audio Proc.*, 13(5):204–217, Sept. 2005.

[18] A. Ben-Yishai and D. Burshtein. A discriminative training algorithm for hidden markov models. *IEEE Trans. Speech and Audio Proc.*, 12(3):204–217, 2004.

[19] M. Bühler, S. Allegro, S. Launer, and N. Dillier. Sound classification in hearing aids inspired by auditory scene analysis. *EURASIP J. App. Signal Proc.*, 18:2991–3002, 2005.

[20] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, USA, 2006.

[21] F. Bonnevier. Audio based context awareness on a pocket pc. M.Sc. thesis, Kungliga Tekniska Högskolan, Stockholm, Sweden, 2006.

[22] G. Boutard, S. Goldszmidt, and G. Peeters. Browsing inside a music track, the experimentation case study. In *Proc. of the 1st Worksop on Learning the Semantics of Audio Signals LSAS 2006*, Athens, Greece, 2006.

[23] J. C. Brown. Calculation of a constant q spectral transform. *J. Acoust. Soc. Am.*, 89(1):425–434, 1991.

[24] J. C. Brown. Determination of the meter of musical scores by autocorrelation. *J. Acoust. Soc. Am.*, 94(4):1953–1957, 1993.

[25] J. C. Brown. Computer identification of musical instruments using pattern recognition with cepstral coefficients as features. *J. Acoust. Soc. Am.*, 105(3), Mar. 1999.

[26] J. C. Brown. Feature dependence in the automatic identification of musical woodwind instruments. *J. Acoust. Soc. Am.*, 109(3), Mar. 2001.

[27] R. Cai, L. Lu, and A. Hanjalic. Co-clustering for auditory scene categorization. *IEEE Trans. Multimedia*, 10(4), 2008.

[28] R. Cai, L. Lu, A. Hanjalic, H.-J. Zhang, and L.-H. Cai. A flexible framework for key audio effects detection and auditory context inference. *IEEE Trans. Audio, Speech, and Language Proc.*, 14(3), May 2006.

[29] P. Cano. *Content-Based Audio Search from Fingerprinting to Semantic Audio Retrieval*. PhD thesis, University Pompeu Fabra, Barcelona, Spain, April 2007.

[30] P. Cano, E. Batlle, T. Kalker, and J. Haitsma. A review of audio fingerprinting. *J. VLSI Signal Proc. Systems*, 41(3):271–284, 2005.

[31] M. Casey. Generalized sound classification and similarity in mpeg-7. *Organized Sound*, 6(2), 2002.

[32] M. Casey, R. Veltcamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. Content-based music information retrieval: Current directions and future challenges. *Proc. IEEE*, 96(4), 2008.

[33] A. T. Cemgil and B. Kappen. Monte carlo methods for tempo tracking and rhythm quantization. *J. Artificial Intelligence Research*, 18:45–81, 2003.

[34] W. Chai. *Automated Analysis of Musical Structure*. PhD thesis, Massachusetts Institute of Technology, MA, USA, September 2005.

[35] B. Clarkson and A. Pentland. Unsupervised clustering of ambulatory audio and video. Tech. report, MIT Media Lab, Perceptual Computing Group.

[36] M. Cooper and J. Foote. Summarizing popular music via structural similarity analysis. In *Proc. IEEE Workshop on Applicat. of Signal Proc. to Audio and Acoust. (WASPAA)*, New Paltz, NY, Oct. 2003.

[37] C. Couvreur, V. Fontaine, P. Gaunard, and C. G. Mubikangiey. Automatic classification of environmental noise events by hidden markov models. *Applied Acoustics*, 54(3):187–206, 1998.

[38] R. B. Dannenberg and N. Hu. Pattern discovery techniques for music audio. In *Proc. Int. Symp. Music Information Retrieval (ISMIR)*, Paris, France, 2002.

[39] M. E. Davies and M. D. Plumbley. Context-dependent beat tracking of musical audio. *IEEE Trans. Audio, Speech, and Language Proc.*, pages 1009–1020, Mar. 2007.

[40] S. B. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Trans. Acoust., Speech, and Signal Proc.*, 28(4):357–366, Aug. 1980.

[41] S. Dixon. Automatic extraction of tempo and beat from expressive performances. *J. New Music Research*, 30(1):39–58, 2001.

[42] S. Dixon. Evaluation of the audio beat tracking system beatroot. *J. New Music Research*, 36(1):39–50, 2007.

[43] S. Dubnov and X. Rodet. Timbre recognition with combined stationary and temporal features. In *Proc. Int. Comp. Music Conf. (ICMC)*, 1998.

[44] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Interscience, 2001.

[45] D. Eck. A positive-evidence model for classifying rhythmical patterns. Tech. report IDSIA-09-00, IDSIA, Lugano, Switzerland, 2000.

[46] J. Eggink and G. J. Brown. A missing feature approach to instrument identification in polyphonic music. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, Hong Kong, Apr. 2003.

[47] K. El-Maleh, A. Samouelian, and P. Kabal. Frame level noise classification in mobile environments. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, volume 1, pages 237–240, Mar. 1999.

[48] D. P. Ellis. *Prediction-driven Computational Auditory Scene Analysis*. Ph.D. thesis, Massachusetts Institute of Tech., June 1996.

[49] D. P. Ellis. Beat tracking by dynamic programming. *J. New Music Research*, 36(1):51–60, 2007.

[50] D. P. Ellis and K. Lee. Accessing minimal-impact personal audio archives. *IEEE Multimedia*, 13(4):30–38, 2006.

[51] A. Eronen. Automatic musical instrument recognition. M.Sc. thesis, Tampere Univ. of Tech., Tampere, Finland, 2001.

[52] S. Essid, G. Richard, and B. David. Instrument recognition in polyphonic music. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, Philadelphia, PA, USA, Mar. 2005.

[53] S. Essid, G. Richard, and B. David. Musical instrument recognition by pairwise classification strategies. 14(4), July 2006.

[54] J. Foote. Visualizing music and audio using self-similarity. In *Proc. ACM Multimedia*, pages 77–80, Orlando, FL, USA, Nov. 1999.

[55] J. T. Foote. Content-based retrieval of music and audio. In C.-C. J. K. *et al.*, editor, *Multimedia Storage and Archiving Systems II, Proc. of SPIE*, volume 3229, pages 138–147, 1997.

[56] A. Fraser and I. Fujinaga. Toward real-time recognition of acoustic musical instruments. In *Proc. Int. Comp. Music Conf. (ICMC)*, 1999.

[57] I. Fujinaga. Machine recognition of timbre using steady-state tone of acoustic musical instruments. In *Proc. Int. Comp. Music Conf. (ICMC)*, 1998.

[58] I. Fujinaga. Realtime recognition of orchestral instruments. In *Proc. Int. Comp. Music Conf. (ICMC)*, 2000.

[59] T. Fujishima. Real-time chord recognition of musical sound: A system using common lisp music. In *Proc. Int. Comp. Music Conf. (ICMC)*, Beijing, China, 1999.

[60] D. Godsmark and G. J. Brown. A blackboard architecture for computational auditory scene analysis. *Speech Communication*, 27:351–366, 1999.

[61] B. Gold and N. Morgan. *Speech and Audio Signal Processing. Processing and Perception of Speech and Music*. John Wiley & Sons, 2000.

[62] E. Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, University Pompeu Fabra, Barcelona, Spain, July 2006.

[63] E. Gomez. Tonal description of polyphonic audio for music content processing. *INFORMS J. on Computing*, 18(3):294–304, 2006.

[64] M. Goto. Real-time beat tracking for drumless audio signals: Chord change detection for musical decisions. *Speech Communication*, 27(3–4):311–335, 1999.

[65] M. Goto. Analysis of musical audio signals. *Chapter in Computational Auditory Scene Analysis, D. Wang and G. Brown, Eds.*, 2006.

[66] M. Goto. A chorus section detection method for musical audio signals and its application to a music listening station. *IEEE Trans. Audio, Speech, and Language Proc.*, 14(5):1783 – 1794, Sept. 2006.

[67] F. Gouyon, P. Herrera, and P. Cano. Pulse-dependent analyses of percussive music. In *Proc. AES 22nd Int. Conf.*, Espoo, Finland, 2002.

[68] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano. An experimental comparison of audio tempo induction algorithms. *IEEE Trans. Audio, Speech, and Language Proc.*, 14(5):1832–1844, 2006.

[69] S. Hainsworth. Beat tracking and musical metre analysis. In A. Klapuri and M. Davy, editors, *Signal Processing Methods for Music Transcription*, pages 101–129. Springer, New York, NY, USA, 2006.

[70] S. W. Hainsworth and M. D. Macleod. Particle filtering applied to musical tempo tracking. *IEEE Trans. Signal Proc.*, pages 2385–2395, 2004.

[71] S. Handel. Timbre perception and auditory object identification. *Chapter in Moore [128]*, pages 425–460.

[72] M. Helen and T. Virtanen. Probabilistic model based similarity measures for audio query-by-example. In *Proc. IEEE Workshop on Applicat. of Signal Proc. to Audio and Acoust. (WASPAA)*, New Paltz, New York, USA, 2007.

[73] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. on Inf. Systems*, 22(1):5–53, 2004.

[74] P. Herrera, G. Peeters, and S. Dubnov. Automatic classification of musical instrument sounds. *J. New Music Research*, 32:3–21, 2003.

[75] P. Herrera-Boyer, A. Klapuri, and M. Davy. Automatic classification of pitched musical instrument sounds. *Chapter in Signal Processing Methods for Music Transcription, A. Klapuri and M. Davy, Eds.*, 2004.

[76] J. Himberg, J. Mäntyjärvi, and P. Korpipää. Using PCA and ICA for exploratory data analysis in situation awareness. In *Proc. IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems*, pages 127–131, Sept. 2001.

[77] J. Häkkilä. *Usability with context-aware mobile applications. Case studies and design guidelines*. Ph.D. thesis, University of Oulu, 2006.

[78] A. Härmä and M. Karjalainen. WarpTB - matlab toolbox for warped DSP (pre-release). `http://www.acoustics.hut.fi/software/warp/`, Sept. 2000.

[79] A. Härmä, M. Karjalainen, L. Savioja, V. Välimäki, U. K. Laine, and J. Huopaniemi. Frequency-warped signal processing for audio applications. *J. Audio Eng. Soc.*, 48(11):1011–1031, 2000.

[80] J. Huopaniemi. Future of personal audio: Smart applications and immersive communication. In *Proc. 30th AES Int. Conf. on Intelligent Audio Environments,*, pages 1–7, Saariselkä, Finland, 2007.

[81] A. Hyvärinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Trans. Neural Networks*, 10(3):626–634, 1999.

[82] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley, 2001.

[83] T. Jehan. *Creating Music by Listening*. PhD thesis, Massachusetts Institute of Technology, MA, USA, September 2005.

[84] J. H. Jensen, M. G. Christensen, D. P. Ellis, and S. H. Jensen. A tempo-insensitive distance measure for cover song identification based on chroma features. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, pages 2209–2212, Las Vegas, USA, Apr. 2008.

[85] K. Jensen. *Timbre Models of Musical Sounds*. Ph.D. thesis, Department of Computer Science, University of Copenhagen, Copenhagen, Denmark, 1999.

[86] K. Jensen. Multiple scale music segmentation using rhythm, timbre, and harmony. *EURASIP J. Adv. in Signal Proc.*, 2007.

[87] K. Jensen and T. H. Andersen. Beat estimation on the beat. In *Proc. IEEE Workshop on Applicat. of Signal Proc. to Audio and Acoust. (WASPAA)*, New Paltz, NY, USA, 2003.

[88] D. Jurafsky and J. H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, New Jersey, USA, 2000.

[89] I. Kaminskyj. Multi-feature musical instrument sound classifier. In *Proc. Australasian Computer Music Conference*, Queensland Univ. of Technology, July 2000.

[90] I. Kaminskyj and A. Materka. Automatic source identification of monophonic musical instrument sounds. In *Proc. of the IEEE Int. Conf. on Neural Networks*, 1995.

[91] K. Kashino and H. Murase. Music recognition using note transition context. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, volume 6, pages 3593–3596, 1998.

[92] K. Kashino and H. Murase. A sound source identification system for ensemble music based on template adaptation and music stream extraction. *Speech Communication*, 27:337–349, 1999.

[93] K. Kashino, K. Nakadai, T. Kinoshita, and H. Tanaka. Application of bayesian probability network to music scene analysis. In *Proc. of the Int. Joint Conf. on AI, CASA workshop*, 1995.

[94] N. Kern. *Multi-Sensor Context-Awareness for Wearable Computing*. Dr.-Ing. thesis, Darmstadt University of Technology, 2005.

[95] H.-G. Kim, N. Moreau, and T. Sikora. *MPEG-7 audio and beyond – audio content indexing and retrieval*. John Wiley & Sons, England, 2005.

[96] T. Kinoshita, S. Sakai, and H. Tanaka. Musical sound source identification based on frequency component adaptation. In *Proc. of the IJCAI-99 Workshop on Computational Auditory Scene Analysis (CASA9́9)*, 1999.

[97] T. Kitahara, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno. Instrument identification in polyphonic music: Feature weighting to minimize influence of sound overlaps. *EURASIP J. App. Signal Proc.*, 2007.

[98] A. Klapuri. Multiple fundamental frequency estimation by summing harmonic amplitudes. In *7th International Conference on Music Information Retrieval (ISMIR-06)*, Victoria, Canada, 2006.

[99] A. P. Klapuri. *Signal Processing Methods for the Automatic Transcription of Music*. Ph.D. thesis, Tampere Univ. of Tech., 2004.

[100] A. Kocsor and J. Csirik. Fast independent component analysis in kernel feature spaces. In *In Proc. SOFSEM 2001*, pages 271–281, 2001.

[101] P. Korpipää. *Blackboard-based software framework and tool for mobile device context awareness*. D.Sc. thesis, VTT Electronics, 2005.

[102] T. A. Kosonen and A. J. Eronen. Rhythm metadata enabled intra-track navigation and content modification in a music player. In *In Proc. of the 5th Int. Conf. on Mobile and Ubiquitous Multimedia, MUM '06*, 2006.

[103] B. Kostek. *Soft Computing in Acoustics: Applications of Neural Networks, Fuzzy Logic and Rough Sets to Musical Acoustics*. Physica-Verlag, 1999.

[104] B. Kostek. Musical instrument classification and duet analysis employing music information retrieval techniques. *Proc. IEEE*, (4), Apr. 2004.

[105] B. Kostek and A. Czyzewski. Automatic recognition of musical instrument sounds - further developments. In *Proc. of the 110th Audio Eng. Soc. Convention*, Amsterdam, Netherlands, May 2001.

[106] K. V. Laerhoven, K. Aidoo, and S. Lowette. Real-time analysis of data from many sensors with neural networks. In *Proc. of the fifth Int. Symp. on Wearable Computers, ISWC 2001*, pages 115–123, 2001.

[107] E. W. Large. Beat tracking with a nonlinear oscillator. In *Proc. IC-JAI Workshop on Artif. Intell. and Music*, pages 24–31, Montreal, Canada, 1995.

[108] J. Laroche. Efficient tempo and beat tracking in audio recordings. *J. Audio Eng. Soc.*, 51(4):226–233, 2003.

[109] A. Lehtiniemi and J. Seppänen. Evaluating SuperMusic: streaming context aware mobile music service. In *Proc. of the Int. Conf. on Adv. in Comp. Entertainment Tech.*, Yokohama, Japan, 2008.

[110] P. Leveau, D. Sodoyer, and L. Daudet. Automatic instrument recognition in a polyphonic mixture using sparse representations. In *Proc. Int. Symp. Music Information Retrieval (ISMIR)*, Vienna, Austria, Sept. 2007.

[111] M. Levy and M. Sandler. Structural segmentation of musical audio by constrained clustering. *IEEE Trans. Audio, Speech, and Language Proc.*, 16(2):318–326, 2008.

[112] M. Levy, M. Sandler, and M. Casey. Extraction of high-level musical structure from audio data and its application to thumbnail generation. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, pages 13–16, 2006.

[113] D. Li, I. Sethi, N. Dimitrova, and T. McGee. Classification of general audio data for content-based retrieval. (22):533–544, 2001.

[114] D. Little and B. Pardo. Learning musical instruments from mixtures of audio with weak labels. In *Proc. Int. Symp. Music Information Retrieval (ISMIR)*, Philadelphia, PA, USA, Sept. 2008.

[115] A. A. Livshin, G. Peeters, and X. Rodet. Studies and improvements in automatic classification of musical sound samples. In *Proc. Int. Computer Music Conference (ICMC 2003)*, Singapore, 2003.

[116] A. A. Livshin and X. Rodet. Musical instrument identification in continuous recordings. In *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Naples, Italy, Oct. 2004.

[117] B. Logan and S. Chu. Music summarization using key phrases. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, pages 749–752, Istanbul, Turkey, May 2000.

[118] L. Lu, M. Wang, and H.-J. Zhang. Repeating pattern discovery and structure analysis from acoustic music data. In *Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval*, pages 275–282, New York, NY, USA, 2004.

[119] L. Lu, H.-J. Zhang, and H. Jiang. Content analysis for audio classification and segmentation. *IEEE Trans. Speech and Audio Proc.*, 10(7):504–516, Oct. 2002.

[120] L. Ma, B. Milner, and D. Smith. Acoustic environment classification. *ACM Transactions on Speech and Language Processing*, 3(2):1–22, July 2006.

[121] N. Maddage. Automatic structure detection for popular music. *IEEE Multimedia*, 13(1):65–77, Jan.–March 2006.

[122] B. Manjunath, P. Salembier, and T. S. (Editors). *Introduction to MPEG-7 Multimedia Content Description Interface*. John Wiley & Sons, England, 2002.

[123] J. Marques and P. J. Moreno. A study of musical instrument classification using gaussian mixture models and support vector machines. CRL 99/4, Compaq Corporation, Cambridge Research laboratory, June 1999.

[124] K. D. Martin. *Sound-Source Recognition: A Theory and Computational Model*. Ph.D. thesis, Massachusetts Institute of Tech., June 1999.

[125] K. D. Martin and Y. E. Kim. Musical instrument identification: A pattern-recognition approach. In *136th meeting of the Acoustical Society of America*, Oct. 1998.

[126] J. McAuley. *On the Perception of Time as Phase: Toward an Adaptive-oscillator Model of Rhythm*. Ph.D. thesis, Indiana Univ., Bloomington, IN, USA, 1995.

[127] M. F. McKinney and D. Moelants. Extracting the perceptual tempo from music. In *Proc. of the 5th Int. Conf. on Music Information Retrieval*, Barcelona, Spain, 2004.

[128] B. C. Moore, editor. *Hearing*. Handbook of Perception and Cognition. Academic Press, San Diego, CA, USA, 2nd edition, 1995.

[129] T. P. Moran and P. Dourish. Introduction to this special issue on context-aware computing. *Human Computer Interaction*, 16(2):87–95, Feb. 2001.

[130] M. D. Nicolas Chetry and M. Sandler. Musical instrument identification using lsf and k-means. In *118th Convention of the AES*, Barcelona, Spain, May 2005.

[131] B. S. Ong. *Structural Analysis and Segmentation of Music Signals*. PhD thesis, University Pompeu Fabra, Barcelona, Spain, February 2007.

[132] E. Pampalk. *Computational Models of Music Similarity and their Application in Music Information Retrieval*. PhD thesis, Vienna University of Technology, Vienna, Austria, March 2006.

[133] T. H. Park and P. Cook. Radial/elliptical basic function neural networks for timbre classification. In *Journees d'informatique musicale*, Maison des Sciences de lHomme Paris Nord, Paris, France, June 2005.

[134] R. Parncutt. A perceptual model of pulse salience and metrical accent in musical rhythms. *Music Perception*, 11(4):409–464, 1994.

[135] J. Paulus and A. Klapuri. Acoustic features for music piece structure analysis. In *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Espoo, Finland, Sept. 2008.

[136] J. Paulus and A. Klapuri. Music structure analysis using a probabilistic fitness measure and an integrated musicological model. In *Proc. of the 9th International Conference on Music Information Retrieval (ISMIR 2008)*, Philadelphia, Pennsylvania, USA, 2008.

[137] G. Peeters. Automatic classification of large musical instrument databases using hierarchical classifiers with inertia ratio maximization. In *115th Convention of the AES*, New York, NY, USA, Oct. 2003.

[138] G. Peeters. Musical key estimation of audio signal based on hidden markov modeling of chroma vectors. In *In Proc. of the 9th Int. Conf. on Digital Audio Effects, DAFx-06*, Montreal, Canada, Sept. 2006.

[139] G. Peeters. Sequence representation of music structure using higher-order similarity matrix and maximum-likelihood approach. In *8th International Conference on Music Information Retrieval (ISMIR-07)*, Vienna, Austria, 2007.

[140] G. Peeters. Template-based estimation of time-varying tempo. *EURASIP J. Adv. in Signal Proc.*, (1):158–171, Jan. 2007.

[141] G. Peeters, A. L. Burthe, and X. Rodet. Toward automatic music audio summary generation from signal analysis. In *In Proc. of the 3rd International Conference on Music Information Retrieval, ISMIR 2002*, Paris, France, 2002.

[142] G. Peeters, S. McAdams, and P. Herrera. Instrument sound description in the context of MPEG-7. In *Proc. Int. Comp. Music Conf. (ICMC)*, Berlin, Germany, 2000.

[143] G. Peeters and X. Rodet. Hierarchical gaussian tree with inertia ratio maximization for the classification of large musical instrument databases. In *Proc. Int. Conf. Digital Audio Effects (DAFx)*, London, UK, Sept. 2003.

[144] V. T. K. Peltonen, A. J. Eronen, M. P. Parviainen, and A. P. Klapuri. Recognition of everyday auditory scenes: Potentials, latencies and cues. In *in Proc. of the 110th Convention of the AES*, Amsterdam, The Netherlands, 2001.

[145] M. Perttunen, M. V. Kleek, O. Lassila, and J. Riekki. Auditory context recognition using svms. In *Proc. UBICOMM 2008*, 2008.

[146] I. Potamitis, N. Fakotakis, and G. Kokkinakis. Independent component analysis applied to feature extraction for robust automatic speech recognition. *Electr. Letters*, Nov. 2000.

[147] H. Purwins, B. Blankertz, and K. Obermayer. A new method for tracking modulations in tonal music in audio data format. In *Proc. IEEE Int. Joint Conf. on Neural Network*, volume 6, pages 270–275, Austin, TX, USA, 2000.

[148] T. F. Quatieri. *Discrete-Time Speech Signal Processing: Principles and Practice*. Prentice Hall, 2002.

[149] L. Rabiner and B.-H. Juang. *Fundamantals of Speech Recognition*. Prentice Hall, 1993.

[150] C. Raphael. Automated rhythm transcription. In *Proc. Int. Symp. Music Information Retrieval (ISMIR)*, pages 99–107, Bloomington, IN, USA, Oct. 2001.

[151] C. Raphael. Musical accompaniment systems. *Systems Chance Magazine*, 17(4):17–22, 2004.

[152] C. Rhodes, M. Casey, S. Abdallah, and M. Sandler. A markov-chain monte-carlo approach to musical audio segmentation. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, pages 797–800, 2006.

[153] D. F. Rosenthal. *Machine Rhythm: Computer Emulation of Human Rhythm Perception*. Ph.D. thesis, Massachusetts Institute of Tech., Aug. 1992.

[154] F. Salam and G. Erten. Sensor fusion by principal and independent component decomposition using neural networks. In *Proc. IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems*, pages 211–215, Aug. 1999.

[155] N. Sawhney. Awareness from environmental sounds. Project report, Speech Interface Group, MIT Media Lab, June 1997.

[156] E. Scheirer and M. Slaney. Construction and evaluation of a robust multifeature speech/music discriminator. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, volume 2, pages 1331–1334, Munich, Germany, Apr. 1997.

[157] E. D. Scheirer. Tempo and beat analysis of acoustic musical signals. *J. Acoust. Soc. Am.*, 103(1):588–601, Jan. 1998.

[158] E. D. Scheirer. Using musical knowledge to extract expressive performance information from audio recordings. In D. Rosenthal and H. Okuno, editors, *Computational Auditory Scene Analysis*. Lawrence Erlbaum Associates, Mahwah, NJ, USA, 1998.

[159] C. E. Schmid. *Acoustic Pattern Recognition of Musical Instruments*. Ph.D. thesis, University of Washington, Washington, USA, 1977.

[160] J. Seppänen. Computational models of musical meter recognition. M.Sc. thesis, Tampere Univ. of Tech., Tampere, Finland, 2001.

[161] J. Seppänen. Tatum grid analysis of musical signals. In *Proc. IEEE Workshop on Applicat. of Signal Proc. to Audio and Acoust. (WASPAA)*, pages 131–134, New Paltz, NY, USA, Oct. 2001.

[162] J. Seppänen and J. Huopaniemi. Interactive and context-aware mobile music experiences. In *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Espoo, Finland, Sept. 2008.

[163] K. Seyerlehner, G. Widmer, and D. Schnitzer. From rhythm patterns to perceived tempo. In *8th International Conference on Music Information Retrieval (ISMIR-07)*, Vienna, Austria, 2007.

[164] R. N. Shephard. Circularity in judgments of relative pitch. *J. Acoust. Soc. Am.*, 36(12):2346–2353, 1964.

[165] Y. Shiu and C.-C. J. Kuo. Musical beat tracking via kalman filtering and noisy measurements selection. In *Proc. IEEE Int. Symp. Circ. and Syst.*, pages 3250–3253, May 2008.

[166] P. Somervuo. Experiments with linear and nonlinear feature transformations in hmm based phone recognition. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, 2003.

[167] F. K. Soong and A. E. Rosenberg. On the use of instantaneous and transitional spectral information in speaker recognition. *IEEE Trans. Acoust., Speech, and Signal Proc.*, 36(6):871–879, June 1988.

[168] H. W. Strube. Linear prediction on a warped frequency scale. *J. Acoust. Soc. Am.*, 68(4):1071–1076, 1980.

[169] P. Toiviainen. An interactive MIDI accompanist. *Comp. Music J.*, 22(4):63–75, 1998.

[170] P. Toiviainen and T. Eerola. Autocorrelation in meter induction: The role of accent structure. *J. Acoust. Soc. Am.*, 119(2):1164–1170, 2006.

[171] G. Tzanetakis and P. Cook. Multifeature audio segmentation for browsing and annotation. In *Proc. IEEE Workshop on Applicat. of Signal Proc. to Audio and Acoust. (WASPAA)*, pages 103–106, New Paltz, NY, USA, 1999.

[172] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Trans. Speech and Audio Proc.*, 10(5):293–302, 2002.

[173] C. Uhle and J. Herre. Estimation of tempo, micro time and time signature from percussive music. In *Proc. 6th Int. Conf. Digital Audio Effects (DAFx-03)*, London, UK, Sept. 2003.

[174] E. Vincent and M. D. Plumbley. Low bit-rate object coding of musical audio using bayesian harmonic models. *IEEE Trans. Audio, Speech, and Language Proc.*, 15(4):1273–1282, 2007.

[175] A. Wang. The shazam music recognition service. *Comm. ACM*, 49(8):44–48, 2006.

[176] D. Wang and G. J. Brown. *Computational Auditory Scene Analysis*. IEEE Press, Piscataway, NJ, USA, 2006.

[177] Y. Wang and M. Vilermo. A compressed domain beat detector using mp3 audio bitstreams. In *Proc. of the Ninth ACM Int. Conf. on Multimedia*, pages 194–202, Ottawa, Canada, 2001.

[178] J. Wellhausen and H. Crysandt. Temporal audio segmentation using MPEG-7 descriptors. In *In Proc. of the SPIE Int. Symp. on ITCom 2003 - Internet Multimedia Management Systems IV*, Orlando (FL), USA, Sept. 2003.

[179] B. A. Whitman. *Learning the Meaning of Music*. Ph.D. thesis, Massachusetts Institute of Tech., June 2005.

[180] G. Wood and S. O'Keefe. On techniques for content-based visual annotation to aid intra-track music navigation. In *Proc. of the 6th Int. Conf. on Music Information Retrieval*, pages 58–65, 2005.

[181] T. Zhang and C.-C. J. Kuo. *Content-based audio classification and retrieval for audiovisual data parsing*. Kluwer Academic Publishers, Dec. 2000.

Author's contribution to the publications

In [P1], the author implemented the algorithms and was the main author of the paper. Prof. Klapuri supervised the research and helped to write the paper.

Publications [P2], [P3], and [P8] were written by the author alone. Valuable comments on draft versions of the papers were received from Prof. Klapuri to publications [P2], [P3] and [P8], and from Jouni Paulus to [P8].

In publication [P4], the author implemented the code related to linear feature transforms and hidden Markov model based tests and run most of the simulations. The largest part of the paper was written by the author. Vesa Peltonen performed and documented the feature comparison experiments. The human perception experiment was carried out and the related section written by Gaëtan Lorho, Timo Sorsa, and Seppo Fagerlund. The other authors participated in various parts of the project and helped in the writing process.

The author had a significant role in formulating the probabilistic model and writing the related part of the publication [P5]. Prof. Klapuri was the main author of the paper, and designed and implemented the method. Prof. Astola helped in the final formulation of the model.

In publication [P6], the author designed the back-end part of the system starting from beat and tatum period estimation and wrote the related parts of the code and paper. The author run the performance simulations and wrote the related description. Jarno Seppänen and Jarmo Hiipakka designed and documented the front-end part of the system.

The ideas in publication [P7] were developed jointly by the author and Prof. Klapuri. The author implemented most of the code and wrote the publication. Prof. Klapuri helped to improve the presentation in the paper.

Errata and Clarifications for the Publications

5.3 Publication [P1]

In Chapter 3: ”Traditionally, the features provided by the timbre research can be divided into spectral and temporal ones. In instrument recognition systems reported so far, only features of either type have been used.” The latter sentence is not correct. At that point, earlier research had used both spectral and temporal features, see e.g. [85, 103]. But to our knowledge, none of the systems had combined cepstral coefficients with other spectral and temporal features, which is proposed in the paper.

In Figure 3, the saxophones are erroneously depicted as brass instruments. Although nowadays made of brass, the saxophones are single reed instruments with a conical bore. The family classification results in Table 2 are also done with saxophones in the brass family. This does not change the conclusions based on the paper.

5.4 Publication [P6]

In the Abstract, the sentence ”Complexity evaluation showed that the computational cost is less than 1% of earlier methods.” should be changed to ”Complexity evaluation showed that the computational cost is less than 1% of two earlier methods.”.

5.5 Publication [P8]

In the Introduction, the sentence ”Similarity-matrix based approaches include the ones by Wellhausen & Crysandt [5] and Cooper & Foote [6].” should be changed to ”An example of a similarity-matrix based approach is the one by Wellhausen & Crysandt [5]”. The Cooper & Foote method should be categorized as ”state” approach, see 3.2.

Publication 1

A. Eronen, A. Klapuri, "Musical Instrument Recognition Using Cepstral Coefficients and Temporal Features", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2000*, pp. 753–756, Istanbul, Turkey, June 2000.

MUSICAL INSTRUMENT RECOGNITION USING CEPSTRAL COEFFICIENTS AND TEMPORAL FEATURES

Antti Eronen and Anssi Klapuri

Signal Processing Laboratory, Tampere University of Technology
P.O.Box 553, FIN-33101 Tampere, FINLAND
eronen@cs.tut.fi, klap@cs.tut.fi

ABSTRACT

In this paper, a system for pitch independent musical instrument recognition is presented. A wide set of features covering both spectral and temporal properties of sounds was investigated, and their extraction algorithms were designed. The usefulness of the features was validated using test data that consisted of 1498 samples covering the full pitch ranges of 30 orchestral instruments from the string, brass and woodwind families, played with different techniques. The correct instrument family was recognized with 94% accuracy and individual instruments in 80% of cases. These results are compared to those reported in other work. Also, utilization of a hierarchical classification framework is considered.

1. INTRODUCTION

Music content analysis in general has many practical applications, including e.g. structured coding, database retrieval systems, automatic musical signal annotation, and musicians' tools. A subtask of this, automatic musical instrument identification, is of significant importance in solving these problems, and is likely to provide useful information also in other sound source identification applications, such as speaker recognition. However, musical signal analysis has has not been able to attain as much commercial interest as, for instance, speaker and speech recognition. This is because the topics around speech processing are more readily commercially applicable, although both areas are considered as being highly complicated.

First attempts in musical instrument recognition operated with a very limited number of instruments and note ranges. De Poli and Prandoni used mel-frequency cepstrum coefficients calculated from isolated tones as an inputs to a Kohonen self-organizing map, in order to construct timbre spaces [2]. Kaminsky and Materka used features derived from an rms-energy envelope and used a neural network or a k-nearest neighbour classifier to classify guitar, piano, marimba and accordion tones over a one-octave band [5].

The recent systems have already shown a considerable level of performance, but have still been able to cope with only a quite limited amount of test data. In [7], Martin reported a system that operates on single isolated tones played over the full pitch ranges of 15 orchestral instruments and uses a hierarchical classification framework. Brown [1] and Martin [8] have managed to build classifiers that are able to operate on test data that include samples played by several different instruments of a particular instrument class, and recorded in environments which are noisy and reverberant. However, even the recent systems are characterized either by a limited application context or by a rather unsatisfactory performance.

In this paper, we aim at utilizing a widest range of features characterizing the different properties of sounds. This is done in order to handle a certain defect in the earlier proposed systems: failure to make simultaneous and effective use of both spectral and temporal features, which is suggested by the work in psychoacoustics. Signal processing methods were implemented that attempt to extract cues about the temporal development, modulation properties, irregularities, formant structure, brightness, and spectral synchronicity of sounds. Although all the factors in sound source identification, and especially their interrelations are not known, a large number of them have been proposed. Thus it looked particularly attractive for us to utilize as much as possible of that information simultaneously in a recognition system, and to see if that would allow us to build a more robust instrument recognition system than described in experiments so far.

Our current implementation handles the isolated tone condition well, and we are hoping that it will generalize to still more realistic contexts. A practical goal of our research is to build an instrument recognition module that can be integrated to an automatic transcription system [6].

This paper is organized as follows. In Section 2, we shortly review the literature in sound source identification and perception. In Section 3, we first take a look at the features used in instrument recognition systems and discuss the approach taken in this paper. Then we describe our feature extraction algorithms. In Section 4, the selected features are validated with thorough simulations and the classification results are compared to those of earlier studies.

2. DIMENSIONS OF TIMBRE

A considerable amount of effort has been done in order to find the perceptual dimensions of *timbre*, the 'colour' of a sound. Often these studies have involved multidimensional scaling experiments, where a set of sound stimuli is presented to subjects, who then give a rating to their similarity or dissimilarity. On the basis of these judgements a low-dimensional space, which best accommodates the similarity ratings, is constructed and a perceptual or acoustic interpretation is searched for these dimensions.

Two of the main dimensions described in these experiments have usually been spectral centroid and rise time [3][9]. The first measures the spectral energy distribution in the steady state portion of a tone, which corresponds to perceived brightness. The second is the time between the onset and the instant of maximal amplitude.

Table 1: Feature descriptions	
1	Rise time, i.e., the duration of attack
2	Slope of line fitted into rms-energy curve after attack
3	Mean square error of line fit in 2
4	Decay time
5	Time between the end of attack and the maximum of rms-energy
6	Crest factor, i.e., *max / rms* of amplitude
7	Maximum of normalized spectral centroid
8	Mean of normalized spectral centroid
9	Mean of spectral centroid
10	Standard deviation of spectral centroid
11	Standard deviation of normalized spectral centroid
12	Frequency of amplitude modulation, range 4-8Hz
13	Strength of amplitude modulation, range 4-8Hz
14	Heuristic strength of the amplitude modulation in range 4-8Hz
15	Frequency of amplitude modulation, range 10-40Hz
16	Strength of amplitude modulation, range 10-40Hz
17	Standard deviation of rise times at each Bark band
18	Mean error of the fit between each of steady state intensities and mean steady state intensity
19	Mean error of fit between each of onset intensities and mean onset intensity
20	Overall variation of intensities at each band
21	Fundamental frequency
22	Standard deviation of fundamental frequency
23-33	Average cepstral coefficients during onset
34-44	Average cepstral coefficients after onset

The psychophysical meaning of the third dimension has varied, but it has often related to temporal variations or irregularity in the spectral envelope. A good review over the enormous body of timbre perception literature can be found in [4]. These available results provide a good starting point for the search of features to be used in musical instrument recognition systems.

3. CALCULATION OF FEATURES

Traditionally, the features provided by the timbre research can be divided into spectral and temporal ones. In instrument recognition systems reported so far, only features of either type have been used. For instance, Kaminsky and Materka used temporal features derived from a short time rms-energy envelope [5]. In the research of Martin [7][8], a selection of temporal features calculated from the outputs of a log-lag correllogram was used, but the spectral shape was not considered at all. Brown reports good results been achieved with cepstral coefficients calculated from oboe and saxophone samples [1]. She used mel-frequency cepstrum coefficients from 23 ms frames, which were then grouped into one or three clusters.

We wanted to test if combining the two types of features, cepstral coefficients and temporal features, would yield the necessary extra discriminating power needed for instrument recognition with a wider set of instruments. The feature set we used is presented in Table 1.

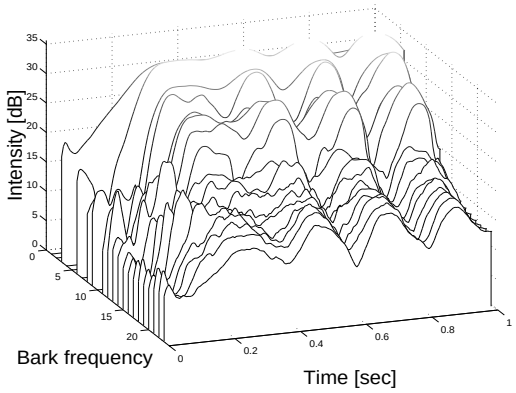


Figure 1. Flute tone: intensities as a function of Bark frequency. Especially amplitude modulation can be seen clearly.

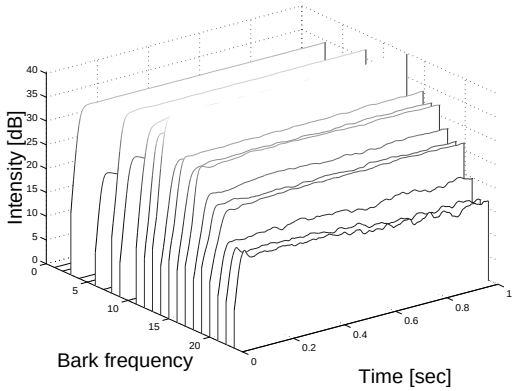


Figure 2. Clarinet tone: intensities as a function of Bark frequency plot. At the low end of clarinet playing range the odd partials are much stronger than the even partials.

3.1 Feature extraction methods

The short-time rms-energy envelope contains information especially about the duration of excitation. We estimated rise-time, decay-time, strenght and frequency of amplitude modulation, crest factor and detected exponential decay from the rms-energy curve calculated in 50% overlapping 10ms frames.

The spectral centroid of the signal is calculated over time in 20ms windows. At each window, the rms-energy of the spectrum is estimated using logarithmic frequency resolution. After that, the spectral centroid is calculated. We use both the absolute value of spectral centroid and a normalized value, which is the absolute value divided by the fundamental frequency. The fundamental frequency estimation method used here is the one presented by Klapuri in [6].

Sinusoid track representation provides many useful temporal features. We first calculate the harmonic amplitude on each of Bark scale bands, which resemble the frequency resolution of the cochlea. Knowledge about the fundamental frequency is applied in

order to resolve whether any harmonics are found on each band. The amplitude envelopes of single harmonic frequencies can be calculated efficiently with an *O(n)* algorithm, where *n* is the sample length. If more than one harmonic frequencies are found, then amplitude envelopes are calculated separately and the resulting band-amplitude is the mean of these. The band-wise intensity is calculated by multiplying the amplitude by the center frequency of the band.

The intensities are decimated by a factor of about 5ms to ease the following computations and smoothed by convolving with a 40ms half-hanning (raised-cosine) window. This window preserves sudden changes, but masks rapid modulation. Figures 1 and 2 display intensity versus Bark frequency plots for 261Hz tones produced by flute and clarinet, respectively.

When the intensity matrix is calculated, a number of features can be easily extracted. The similarity of shape between intensity envelopes is measured by fitting the envelopes into a mean envelope and calculating the mean of mean squre errors. This is done separately for the onset period and the rest of the waveform. The error value of the onset period, accompanied with the standard deviation of bandwise rise times, can be considered as a measure of onset asynchrony. Another measure that can be extracted from the intensity envelope curves is the overall variation of intensities at each band.

The spectral shape of tones is modelled with cepstral coefficients, which are calculated with a method adapted from an automatic speech recognition system described in [11]. Calculation procedure is done in 25% overlapping windowed frames of size approximately 20ms. Autocorrelation sequence is calculated first and then used for LPC coefficient calculation with Levinson-Durbin algorithm. LPC coefficients are then converted into cepstral coefficients, which have been found to be a robust feature set for use in speech and instrument recognition [1]. We used two sets of 11 coefficients, averaged over the onset and the rest of the sample.

4. CLASSIFICATION

Musical instruments form a natural hierarchy, which includes different instrument families. In many applications, classification down to the level of instrument families is sufficient for practical needs. For example, searching a database to find string music would make sense. In addition to that, a classifier may utilize a hierarchical structure algorithmically while assigning a sound into a lowest level class, individual instrument. This has been proposed and used by Martin in [7][8]. In the following, we give a short review of his principles. At the top level of the taxonomy, instruments are divided into pizzicato and sustained. Second level comprises instrument families, and the bottom level individual instruments. Classification occurs at each node, applying knowledge of the best features to distinguish between possible subclasses. This way of processing is suggested to have some advantages over direct classification at the lowest end of the taxonomy, because the decision process may be simplified to take into account only the small number of possible subclasses.

Table 2: Classification results

	Hierarchy 1	**Hierarchy 2**	**No hierarchy**
Pizzicato / sustained	99.0%	99.0%	99.0%
Instrument families	93.0%	94.0%	94.7%
Individual instruments	74.9%	75.8%	80.6%

In our system, at each node a Gaussian or a k-NN classifier was used with a fixed set of features. The Gaussian classifier turned out to yield the best results at the highest level, where the number of classes is two. At the lower levels, k-NN classifier was used. Bad features are likely to decrease classifying performance, which makes evaluating the salience of each feature essential. The features used at a node were selected manually by monitoring feature values of possible subclasses. This was done one feature at a time, and only the features making clear distinction were included into the feature set of the node.

We implemented a classification hierarchy similar to that presented by Martin in [7], with the exception that his samples and taxonomy did not include piano. In our system the piano was assigned to an own family node because of having a unique set of some features, especially cepstral coefficients. According to Martin, classification performance was better if the reeds and the brass were first processed as one family and separated at the next stage. We wanted to test this with our own feature set and test data and tried the taxonomy with and without the Brass or Reeds node, which is marked with a '*' in Figure 3.

5. RESULTS

Our validation database consisted of 1498 solo tones covering the entire pitch ranges of 30 orchestral instruments with several articulation styles (e.g. pizzicato, martele, bowed, muted, flutter), as illustrated in Figure 3. All tones were from the McGill Master Samples collection [10], except the piano and quitar tones which were played by amateur musicians and recorded with a DAT recorder. In order to achieve comparable results to those described by Martin in [7], similar way of cross validation with 70% / 30% splits of train and test data was used. A difference to the method of Martin was to estimate the fundamental frequency of the test sample before classification, which was then compared to the pitch ranges of different instruments, taking only the possible ones into classification.

In Table 2, we present the classification results made in the three different ways. Hierarchy 1 is the taxonomy of Figure 6 without the Brass or Reeds node. In the No-hierarchy experiment classification was made separately for each classification level. The Hierarchy 2 proved out to yield slightly better results, like Martin reported in [7]. But interestingly, in our experiments, the direct classification at each level performed best at both tasks, which was not the case in Martin's experiments where the Hierarchy 2 yielded the best results. At the current implementation, classification result at the lower level of hierarchy is totally dependent on the results of the higher levels, and the error cumulates as the classification proceeds.

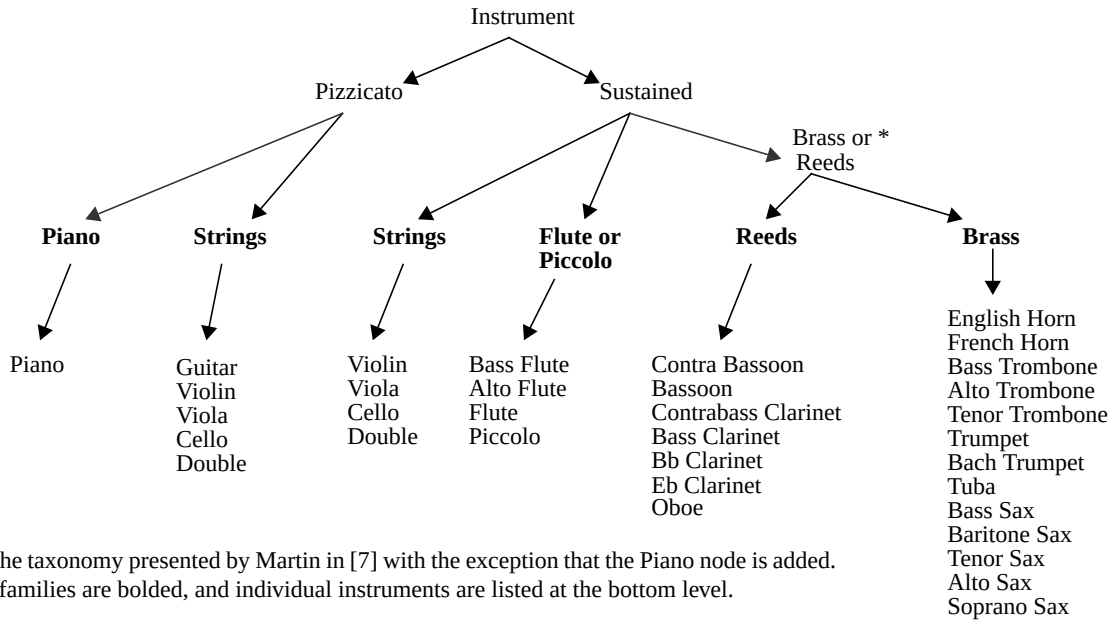


Figure 3. The taxonomy presented by Martin in [7] with the exception that the Piano node is added. Instrument families are bolded, and individual instruments are listed at the bottom level.

No significant advantage was achieved with hierarchical classification. Perhaps the biggest benefit of hierarchical approach would be got if more than one possible choises at each node were taken into account and the salience of the features was automatically evaluated. Classification of this kind has been used in [8].

The achieved performance both in instrument family and individual instrument classification was better than reported by Martin in [7]. His system's classification accuracies were approximately 90% in instrument family and 70% with individual instruments, while the data set consisted 1023 samples of 15 different instruments, being a subset of our data. Comparison to other systems is not reasonable because of the different amount of instruments or different method of performance evaluation used [1][5][8].

6. CONCLUSIONS

A system for musical instrument recognition was presented that uses a wide set of features to model the temporal and spectral charactreristics of sounds. Signal processing algorithms were designed to measure these features in acoustic signals. Using this input data, a classifier was constructed and the usefulness of the features was verified. Furthermore, experiments were carried out to investigate the potential advantage of a hierarchically structured classifier.

The achieved performance and comparison to earlier results demonstrates that combining the different types of features succeeded in capturing some extra knowlege about the instrument properties. Hierarchical structure could not bring further benefits, but its full potential should be reconsidered when a wider data set including more instruments, as well as different examples from a particular instrument class is available. Future work will concentrate on these areas, and on integrating the recognizer into a system that is able to process more complex sound mixtures.

7. REFERENCES

[1] Brown, J. C. "Computer identification of musical instruments using pattern recognition with cepstral coefficients as features." J. Acoust. Soc. Am. 105(3) 1933-1941.

[2] De Poli, G. & Prandoni, P. "Sonological Models for Timbre Characterization". Journal of New Music Research, Vol 26 (1997), pp. 170-197, 1997.

[3] Grey, J. M. "Multidimensional perceptual scaling of musical timbres". J. Acoust. Soc. Am. 61(5), 1270-1277, 1977.

[4] Handel, S. "Timbre perception and auditory object identification". In Moore (ed.) Hearing. New York: Academic Press.

[5] Kaminskyj, I. & Materka, A. "Automatic source identification of monophonic musical instrument sounds". Proceedings of the 1995 IEEE International Conference of Neural Networks, pp. 189-194, 1995.

[6] Klapuri, A. "Pitch Estimation Using Multiple Independent Time-Frequency Windows". Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, New Paltz, New York, Oct. 17-20, 1999.

[7] Martin, K. D. "Musical Instrument Identification: A Pattern Recognition Approach". Presented at the 136th meeting of the Acoustical Society of America, 1998.

[8] Martin, K. D. "Sound-Source Recognition: A Theory and Computational Model". Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, 1999.

[9] McAdams S., Winsberg S., Donnadieu S., De Soete G., Krimphoff, J. "Perceptual scaling of synthesized musical timbres: common dimensions, specifities, and latent subject classes". Psychological Research, 58, pp 177-192, 1995.

[10] Opolko, F. & Wapnick, J. "McGill University Master Samples" (compact disk). McGill University, 1987.

[11] Rabiner, L. R. & Juang, B. H. "Fundamentals of speech recognition". Prentice-Hall 1993.

Publication 2

A. Eronen, "Comparison of features for musical instrument recognition", *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, WASPAA 2001*, pp. 19–22, New Paltz, New York, USA, October 2001.

COMPARISON OF FEATURES FOR MUSICAL INSTRUMENT RECOGNITION

Antti Eronen

Signal Processing Laboratory, Tampere University of Technology
P.O.Box 553, FIN-33101 Tampere, Finland
antti.eronen@tut.fi

ABSTRACT

Several features were compared with regard to recognition performance in a musical instrument recognition system. Both mel-frequency and linear prediction cepstral and delta cepstral coefficients were calculated. Linear prediction analysis was carried out both on a uniform and a warped frequency scale, and reflection coefficients were also used as features. The performance of earlier described features relating to the temporal development, modulation properties, brightness, and spectral synchronity of sounds was also analysed. The data base consisted of 5286 acoustic and synthetic solo tones from 29 different Western orchestral instruments, out of which 16 instruments were included in the test set. The best performance for solo tone recognition, 35% for individual instruments and 77% for families, was obtained with a feature set consisting of two sets of mel-frequency cepstral coefficients and a subset of the other analysed features. The confusions made by the system were analysed and compared to results reported in a human perception experiment.

1. INTRODUCTION

Automatic musical instrument recognition is a fascinating and essential subproblem in music indexing, retrieval, and automatic transcription. It is closely related to computational auditory scene analysis. However, musical instrument recognition has not received as much research interest as speaker recognition, for instance.

The implemented musical instrument recognition systems still have limited practical usability. Brown has reported a system that is able to recognize four woodwind instruments from monophonic recordings with a performance comparable to that of human's [1]. Martin's system recognized a wider set of instruments, although it did not perform as well as human subjects in a similar task [2].

This paper continues the work presented in [3] by using new cepstral features and introducing a significant extension to the evaluation data. The research focuses on comparing different features with regard of recognition accuracy in a solo tone recognition task. First, we analyse different cepstral features that are based either on linear prediction (LP) or filterbank analysis. Both conventional LP having uniform frequency resolution and more psychoacoustically motivated warped linear prediction (WLP) are used. WLP based features have not been used for musical instrument recognition before. Second, other features are analysed that are related to the temporal development, modulation properties, brightness, and spectral synchronity of sounds.

The evaluation database is extended to include several examples of a particular instrument. Both acoustic and synthetic isolated notes of 16 Western orchestral instruments are used for testing, whereas the training data includes examples of 29 instruments. The performance of the system and the confusions it makes are compared to the results reported in a human perception experiment, which used a subset of the same data as stimuli [2].

2. FEATURE EXTRACTION

2.1. Cepstral features

For isolated musical tones, the onset has been found to be important for recognition by human subjects [4]. Motivated by this, the cepstral analyses are made separately for the onset and steady state segments of a tone. Based on the root mean square (RMS) -energy level of the signal, each tone is segmented into onset and steady state segments. The steady state begins when the signal achieves its average RMS-energy level for the first time, and the onset segment is the 10 dB rise before this point.

For the onset portion of tones, both LP and filterbank analyses were performed in approximately 20 ms length hamming windowed frames with 25% overlap. In the steady state segment, frame length of 40 ms was used. If the onset was shorter than 80 ms, the beginning of steady state was moved forward so that at least 80 ms was analysed. Prior to the analyses, each acoustic signal was preemphasized with the high pass filter $1, -0.97z^{-1}$ to flatten the spectrum.

The LP coefficients were obtained from an all-pole approximation of the windowed waveform, and were computed using the autocorrelation method. In the calculation of the WLP coefficients, the frequency warping transformation was obtained by replacing the unit delays of the predicting filter with first-order all-pass elements. In the z-domain this can be interpreted by the mapping

$$z^{-1} \rightarrow \tilde{z}^{-1} = \frac{z^{-1} - \lambda}{1 - \lambda z^{-1}}. \quad (1)$$

In the implementation this means replacing the autocorrelation network with a warped autocorrelation network [5]. The parameter λ is selected in such a way that the resulting frequency mapping approximates the desired frequency scale. By selecting λ=0.7564 for 44.1 kHz samples, a Bark scale approximation was obtained [6]. Finally, the obtained linear prediction coefficients a_n are transformed into cepstral coefficients c_n with the recursion [7, pp. 163]

$$c_n = -a_n - \frac{1}{n}\sum_{k=1}^{n-1} k c_k a_{n-k}. \quad (2)$$

The number of cepstral coefficients was equal to the analysis order after the zeroth coefficient, which is a function of the channel gain, was discarded.

For the mel-frequency cepstral coefficient (MFCC) calculations, a discrete Fourier transform was first calculated for the win-

dowed waveform. The length of the transform was 1024 or 2048 point for 20 ms and 40 ms frames, respectively. 40 triangular bandpass filters having equal bandwith on the mel-frequency scale were simulated, and the MFCCs were calculated from the log filterbank amplitudes using a shifted discrete cosine transform [7, p.189].

In all cases, the median values of cepstral coefficients were stored for the onset and steady state segments. Delta cepstral coefficients were calculated by fitting a first order polynomial over the cepstral trajectories. For the delta-cepstral coefficients, the median of their absolute value was calculated. We also experimented with coefficient standard deviations in the case of the MFCCs.

2.2. Spectral and temporal features

Calculation of the other features analysed in this study has been described in [3] and will be only shortly summarized here.

Amplitude envelope contains information e.g. about the type of excitation; i.e. whether a violin has been bowed or plucked. Tight coupling between the excitation and the resonance structure is indicated by a short onset duration. To measure the slope of the amplitude decay after the onset, a line was fitted over the amplitude envelope on a dB scale. Also, the mean square error of the fit was used as a feature. Crest factor, i.e. maximum / RMS value was also used to characterize the shape of the amplitude envelope.

Strength and frequency of amplitude modulation (AM) was measured at two frequency ranges: from 4-8 Hz to measure tremolo, i.e. AM in conjunction with vibrato, and 10-40 Hz for graininess or roughness of tones.

Spectral centroid (SC) corresponds to perceived brightness and has been one of the interpretations for the dissimilarity ratings in many multidimensional scaling studies [4]. SC was calculated from a short time power spectrum of the signal using logarithmic frequency resolution. The normalized value of SC is the absolute value in Hz divided by the fundamental frequency. The mean, maximum and standard deviation values of SC were used as features.

Onset asynchrony refers to the differences in the rate of the energy development of different frequency components. A sinusoid envelope representation was used to calculate the intensity envelopes for different harmonics, and the standard deviation of onset durations for different harmonics was used as a one feature. Another feature measuring this property is obtained by fitting the intensity envelopes of individual harmonics into the overall intensity evelope during the onset period, and the average mean square error of those fits was used as a feature.

Fundamental frequency (f0) of tones is measured using the algorithm from [8], and used as a feature. Also, its standard deviation was used as measure for vibrato.

3. EXPERIMENTAL SETUP

Samples from five different sources were included in the validation database. First, the samples used in [3] consisted of the samples from the McGill University Master Samples Collection (MUMS) [9], as well as recordings of an acoustic guitar made at Tampere University of Technology. The other sources of samples were the University of Iowa website, IRCAM Studio Online (SOL), and a Roland XP-30 synthesizer. The MUMS and SOL samples are recorded in studios with different acoustic characteristics and recording equipment, and the samples from Iowa University are recorded in an anechoic chamber. The samples from the Roland synthesizer were played on the keyboard and recorded through analog lines into a Silicon Graphics Octane workstation. The synthesizer has a dynamic keyboard, thus these samples have varying dynamics. The samples from SOL include only the first 1.5 seconds of the played note.

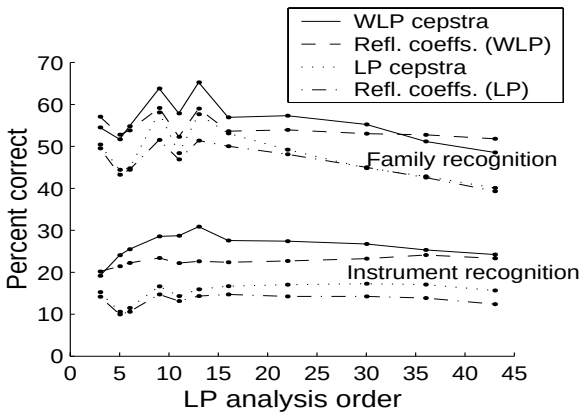


Figure 1. *Classification performance as a function of analysis order for different LP based features.*

Cross validation aimed at as realistic conditions as possible with this data set. On each trial, the training data consisted of all the samples except those of the particular performer and instrument being tested. In this way, the training data is maximally utilized, but the system has never heard the samples from that particular instrument in those circumstances before. There were 16 instruments that had at least three independent recordings, so these instruments were used for testing. The instruments can be seen in Figure 4. A total of 5286 samples of 29 Western orchestral instruments were included in the data set, out of which 3337 samples were used for testing. The classifier made its choice among the 29 instruments. In these tests, a random guesser would score 3.5% in the individual instrument recognition task, and 16.7% in family classification.

In each test, classifications were performed separately for the instrument family and individual instrument cases. A *k*-nearest neighbours (kNN) classifier was used, where the values of *k* were 11 for instrument family and for 5 individual instrument classification. The distance metric was Mahalanobis with equal covariance matrix for all classes, which was implemented by using the discrete form of the Karhunen-Loeve transform to uncorrelate the features and normalize the variances, and then by using the euclidean distance metric in the normalized space.

4. RESULTS

Different orders of the linear prediction filter were used to see the effect of that on the performance of several LP and WLP based features. The results for instrument family and individual instrument recognition are shown in Figure 1. The feature vector at all points consisted of two sets of coefficients: medians over the onset period and medians over the steady state. The optimal analysis order was between 9 and 14, above and below which per-

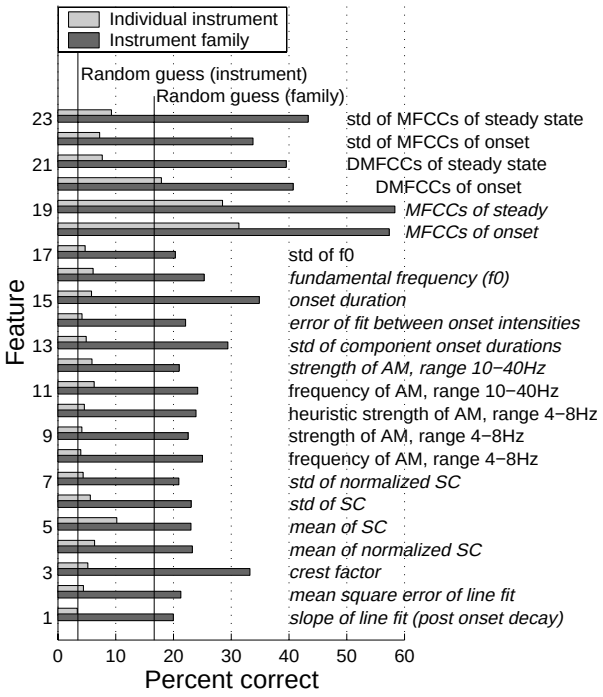


Figure 2. *Classification performance as a function of features. The features printed in italics were included in the best performing configuration.*

formance degrades. The number of cepstral coefficients was one less than the analysis order. WLP cepstral and reflection coefficients outperformed LP cepstral and reflection coefficients at all analysis orders calculated. The best accuracy with LP based features was 33% for individual instruments (66% for instrument families), and was obtained with WLP cepstral coefficients (WLPCC) of order 13.

In Figure 2, the classification accuracy is presented as a function of features. The cepstral parameters are mel-frequency cepstral coefficients or their derivatives. The optimal number of MFCCs was 12, above and below which the performance slowly degraded. However, optimization of the filter bank parameters should be done for the MFCCs, but was left for future research. By using the MFCCs both from the onset and steady state, the accuracies were 32% (69%). Because of computational cost considerations the MFCC were selected as the cepstrum features for the remaining experiments. Adding the mel-frequency delta cepstrum coefficients (DMFCC) slightly improved the performance, using the MFCCs and DMFCCs of the steady state resulted in 34% (72%) accuracy.

The other features did not alone prove out very successful. Onset duration was the most successful with 35% accuracy in instrument family classification. In individual instruments, spectral centroid gave the best accuracy, 10%. Both were clearly inferior to the MFCCs and DMFCCs. It should be noted, however, that the MFCC features are vectors of coefficients, and the other features consist of a single number each.

The best accuracy 35% (77%) was obtained by using a feature vector consisting of the features printed in italics in Figure 2. The feature set was found by using a subset of the data and a simple

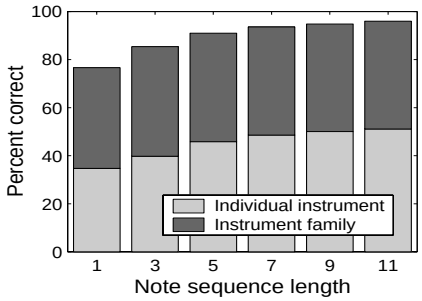


Figure 3. *Classification performance as a function of note sequence length.*

backward select algorithm. If the MFCCs were replaced with order 13 WLPCCs, the accuracy was 35% (72%).

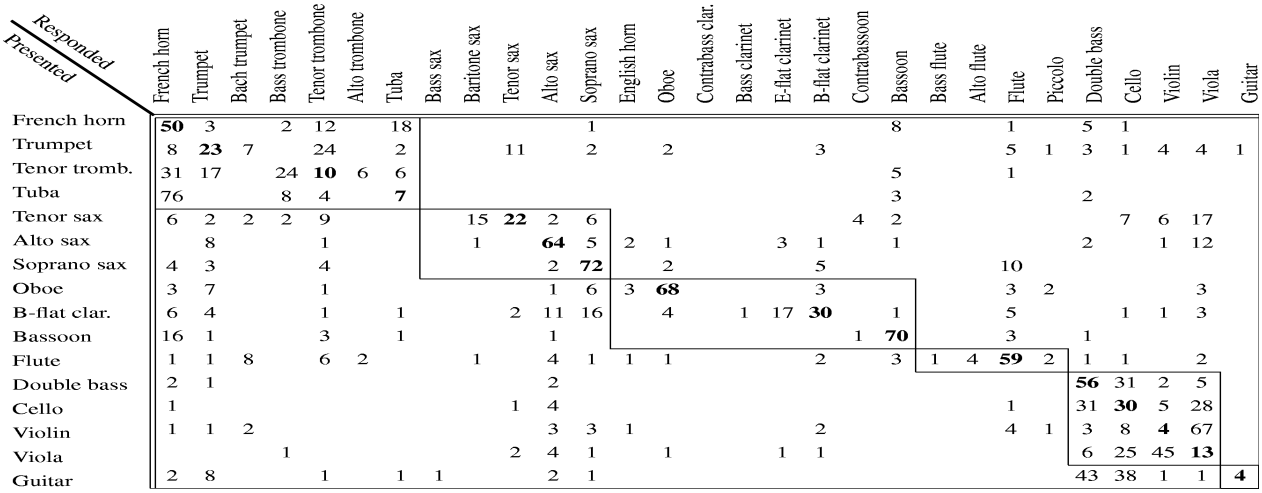
In practical situations, a recognition system is likely to have more than one note to use for classification. A simulation was made to test the system's behaviour in this situation. Random sequences of notes were generated and each note was classified individually. The final classification result was pooled across the sequence by using the majority rule. The recognition accuracies were averaged over 50 runs for each instrument and note sequence length. Figure 3 shows the average accuracies for individual instrument and family classification. With 11 random notes, the average accuracy increased to 51% (96%). In instrument family classification, the recognition accuracy for the tenor saxophone was the worst (55% with 11 notes), whereas the accuracy for the all other instruments was over 90%. In the case of individual instruments, the accuracy for the tenor trombone, tuba, cello, violin, viola and guitar was poorer than with one note, the accuracy for the other instruments was higher.

The recognition accuracy depends on the recording circumstances, as may be expected. The individual instrument recognition accuracies were 32%, 87%, 21% and 37% for the samples from MUMS, Iowa, Roland and SOL sources, respectively. The Iowa samples included only the woodwinds and the French horn, which were on the average recognized with 49% accuracy. Thus, the recognition accuracy is clearly better for the Iowa samples recorded in an anechoic chamber. The samples from the other three sources are comparable with the exception that the samples from SOL did not include tenor or soprano sax. With synthesized samples the performance is clearly worse, which is probably due to both the varying quality of the synthetic tones and the varying dynamics.

5. DISCUSSION

The confusion matrix for the feature set giving the best accuracy is presented in Figure 4. There are large differences in the recognition accuracies of different instruments. The soprano sax is recognized correctly in 72% of the cases, while the classification accuracies for the violin and guitar are only 4%. French horn is the most common target for misclassifications.

It is interesting to compare the behaviour of the system to human subjects. Martin [2] has reported a listening experiment where fourteen subjects recognized 137 samples from the McGill collection, a subset of the data used in our evaluations. The differences in the instrument sets are small, Martin's samples did not



Presented \ Responded	French horn	Trumpet	Bach trumpet	Bass trombone	Tenor trombone	Alto trombone	Tuba	Bass sax	Baritone sax	Tenor sax	Alto sax	Soprano sax	English horn	Oboe	Contrabass clar.	Bass clarinet	E-flat clarinet	B-flat clarinet	Contrabassoon	Bassoon	Bass flute	Alto flute	Flute	Piccolo	Double bass	Cello	Violin	Viola	Guitar
French horn	**50**	3		2	12		18					1								8			1		5	1			
Trumpet	8	**23**	7		24		2			11		2		2				3					5	1	3	1	4	4	1
Tenor tromb.	31	17		24	**10**	6	6													5			1						
Tuba	76			8	4		**7**													3					2				
Tenor sax	6	2	2	2	9				15	**22**	2	6							4	2						7	6	17	
Alto sax		8			1				1		**64**	5	2	1			3	1		1					2		1	12	
Soprano sax	4	3			4						2	**72**		2				5					10						
Oboe	3	7			1						1	6	3	**68**				3					3	2				3	
B-flat clar.	6	4			1		1			2	11	16		4		1	17	**30**		1			5			1	1	3	
Bassoon	16	1			3		1				1								1	**70**			3		1				
Flute	1	1	8		6	2			1		4	1	1	1				2		3	1	4	**59**	2	1	1		2	
Double bass	2	1									2														**56**	31	2	5	
Cello	1									1	4												1		31	**30**	5	28	
Violin	1	1	2								3	3	1					2					4	1	3	8	**4**	67	
Viola				1						2	4	1		1			1	1							6	25	45	**13**	
Guitar	2	8			1		1	1			2	1													43	38	1	1	**4**

Figure 4. *Confusion matrix for the best performing feature set. Entries are expressed as percentages and are rounded to the nearest integer. The boxes indicate instrument families.*

include any sax or guitar samples, but had the piccolo and the English horn, which were not present in our test data. In his test, the subjects recognized the individual instrument correctly in 45.9% of cases (91.7% for instrument families). Our system made more outside family confusions than the subjects in Martin's test. It was not able to generalize into more abstract instrument families as well as humans, which was also the case in Martin's computer simulations [2]. In individual instrument classification, the difference is perhaps smaller.

The within-family confusions made by the system are quite similar to the confusions made by humans. Examples include the French horn as tenor trombone and vice versa, tuba as French horn, or B-flat clarinet as E-flat clarinet. The confusions between the viola and the violin, and the cello and the double bass were also common to both humans and our system. In the confusions occurring outside the instrument family, confusions of the B-flat clarinet as soprano or alto sax were common to both our system and the subjects.

6. CONCLUSIONS

Warped linear prediction based features proved to be successful in the automatic recognition of musical instrument solo tones, and resulted in better accuracy than what was obtained with corresponding conventional LP based features. The mel-frequency cepstral coefficients gave the best accuracy in instrument family classification, and would be the selection also for the sake of computational complexity. The best overall accuracy was obtained by augmenting the mel-cepstral coefficients with features describing the type of excitation, brightness, modulations, synchronity and fundamental frequency of tones.

Care should be taken while interpreting the presented results on the accuracy obtained with different features. First, the best set of features for musical instrument recognition depends on the context [2,4]. Second, the extraction algorithms for features other than cepstral coefficients are still in their early stages of development. However, since the accuracy improved when cepstral features were added with other features, this approach should be further developed.

7. ACKNOWLEDGEMENT

The available samples in the web by the University of Iowa (http://theremin.music.uiowa.edu/~web/) and IRCAM (http://soleil.ircam.fr/) helped greatly in collecting our database. The warping toolbox by Härmä and Karjalainen (http://www.acoustics.hut.fi/software/warp/) was used for the calculation of WLP based features. Our MFCC analysis was based on Slaney's implementation (http://rvl4.ecn.purdue.edu/~malcolm/interval/1998-010/).

8. REFERENCES

[1] Brown, J. C. "Feature dependence in the automatic identification of musical woodwind instruments." *J. Acoust. Soc. Am.*, Vol. 109, No. 3, pp. 1064-1072, 2001.

[2] Martin, K. D. *Sound-Source Recognition: A Theory and Computational Model.* Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, 1999. Available at: http://sound.media.mit.edu/Papers/kdm-phdthesis.pdf.

[3] Eronen, A. & Klapuri, A. "Musical Instrument Recognition Using Cepstral Coefficients and Temporal Features". *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Istanbul, June 5-9, 2000.

[4] Handel, S. *Timbre perception and auditory object identification*. In Moore (ed.) *Hearing*. New York, Academic Press.

[5] Härmä, A. et.al. "Frequency-Warped Signal Processing for Audio Applications". *J. Audio Eng. Soc.*, Vol. 48, No. 11, pp. 1011-1031, 2000.

[6] Smith, J. O. & Abel, J. S. "Bark and ERB Bilinear Transforms". *IEEE Transactions on Speech and Audio Processing*, Vol. 7, No. 6, pp. 697-708, 1999.

[7] Rabiner, L. R. & Juang, B. H. *Fundamentals of speech recognition*. Prentice-Hall 1993.

[8] Klapuri, A. "Pitch Estimation Using Multiple Independent Time-Frequency Windows". *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, New York, Oct. 17-20, 1999.

[9] Opolko, F. & Wapnick, J. *McGill University Master Samples* (compact disk). McGill University, 1987.

Publication 3

A. Eronen, "Musical instrument recognition using ICA-based transform of features and discriminatively trained HMMs", *Proceedings of the Seventh International Symposium on Signal Processing and its Applications, ISSPA 2003*, Vol. 2, pp. 133–136, Paris, France, July 2003.

MUSICAL INSTRUMENT RECOGNITION USING ICA-BASED TRANSFORM OF FEATURES AND DISCRIMINATIVELY TRAINED HMMS

Antti Eronen
Tampere University of Technology, Institute of Signal Processing
P.O. Box 553, FIN-33101 Tampere, Finland
antti.eronen@tut.fi

ABSTRACT

In this paper, we describe a system for the recognition of musical instruments from isolated notes or drum samples. We first describe a baseline system that uses uses mel-frequency cepstral coefficients and their first derivatives as features, and continuous-density hidden Markov models (HMMs). Two improvements are proposed to increase the performance of this baseline system. First, transforming the features to a base with maximal statistical independence using independent component analysis can give an improvement of 9 percentage points in recognition accuracy. Secondly, discriminative training is shown to further improve the recognition accuracy of the system. The evaluation material consists of 5895 isolated notes of Western orchestral instruments, and 1798 drum hits.

1 INTRODUCTION

Earlier work on musical instrument recognition has mainly used classifiers that are not able to effectively model the temporal evolution of spectral features. The Gaussian mixture model (GMM) ([1]) is able to effectively parameterize the distribution of observations. However, it does not explicitly model the dynamic evolution of feature values within a played note. One approach is to extract features that explicitly try to measure the temporal characteristics of isolated notes [2], or to manually segment the notes and use averages of cepstral coefficients during the onset (the beginning of a note) and steady state as features [3]. However, this has only a limited ability to model the temporal evolution even if feature variances were also used as features. Moreover, often the extraction of temporal features is computationally rather demanding and the effect is even greater if this is combined with the use of a nearest-neighbour classifier, for instance.

Hidden Markov models (HMM) are the mainstream statistical model used in the speech recognition community, and are now becoming increasingly popular also in non-speech applications. To our knowledge, Casey is the only researcher who has used HMMs to model musical instrument samples [4]. As a part of the development of the generalized audio descriptors for the MPEG-7 standard, he has evaluated the proposed methods using a database consisting of a wide variety of audio, including music, speech, environmental sounds, and different musical instrument sounds.

However, Casey's evaluation data has included examples of only a few instruments. In addition, little detail has been given on the difficulty of the evaluation material, making assessing the accuracy of his method in instrument recognition difficult. Moreover, no details were given on the topology of the resulting models, since their algorithm attempts to force some of the transition probabilities to zero during training [4].

In this paper, we take a different approach. Based on the knowledge of physical properties of musical instruments, and on the other hand the psychological studies on timbre perception, there is a clear motivation for using HMMs with a left-right topology to model isolated notes. Most musical instruments have a distinctive onset period, followed by a steady state, and finally decay (or release). For instance, some instruments are characterized by onset asynchrony, which means that the energy of certain harmonics rises more quickly than the energy at some other frequencies. Also the decay is often characterised by the prominence of certain frequencies with respect to others. This causes the features relating to the spectral shape to have different value distributions during the onset, steady state, and decay. Thus, a left-right HMM with three states might well model this temporal evolution.

This paper first describes the development of a baseline instrument recognizer that uses mel-frequency cepstrum (MFCC) and delta cepstrum (ΔMFCC) coefficients as features, and HMMs to model the feature distributions. The system is evaluated using a database consisting of isolated notes of 27 Western orchestral instruments, and a smaller database of drum hits. We propose two improvements to improve the performance of the system. First, we use the independent component analysis (ICA) to transform the feature vector consisting of catenated MFCC and ΔMFCC features to a basis with maximal statistical independence. This transform is shown to give an almost consistent improvement in recognition accuracy over the baseline with no rotation. Second, we propose using discriminative training of the HMMs. Especially with computationally attractive models with low number of components in state densities, discriminative training gives an improvement over the baseline maximum likelihood (ML) training using the Baum-Welch re-estimation algorithm.

2 FEATURE EXTRACTION

2.1 Feature extraction

Mel-frequency cepstral coefficients (MFCC) were found to be a well-performing feature set in musical instrument recognition [3], and are used as the front-end parameters in

our system. The input signal is first pre-emphasized with an FIR filter having the transfer function $1-az^{-1}$, where a was between 0.97 and 0.99 in our simulations. MFCC analysis is performed in 30 ms windowed frames advanced every 15 ms for the orchestral instruments. For the analysis of short drum sounds, the frame length was reduced into 20 ms, and the hop size was 4 ms. The number of triangular filters was 40, and they occupied the band from 30Hz to half the sampling rate. For the drum sounds, the lowest frequency was 20Hz. The number of cepstral coefficients was 12 after the zeroth coefficient was discarded, and appending the first time derivatives approximated with a 3-point first-order polynomial fit resulted in a feature vector size of n = 24. The resulting features were both mean and variance normalized.

2.2 Transforming features using independent component analysis (ICA)

Independent component analysis (ICA) has recently emerged as an interesting method for finding decorrelating feature transformations [4][5][6]. The more well-known methods for include the principal component analysis and linear discriminant analysis. The goal of ICA is to find directions of minimum mutual information, i.e. to extract a set of statistically independent vectors from the training data **X**. The use of an ICA transformation has been reported to improve the recognition accuracy in speech recognition [5]. In the MPEG-7 generalized audio descriptors, ICA is proposed as an optional transformation on the spectrum basis obtained with singular value decomposition [4], and Casey's results have shown the success of this method on a wide variety of sounds. Our approach is slightly different from all these studies. We perform ICA on concatenated MFCC and ΔMFCC features. In [4] and [5] only static features were used, and in [6] logarithmic energies and their derivatives were used.

In order to construct the m-by-n ICA transform matrix **W**, the extracted MFCC and ΔMFCC coefficients from the training data samples are gathered into a matrix $\mathbf{X}=[\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_T]$ where each column represents the catenated MFCC (s) and ΔMFCC (d) features from the analysis frame t, i.e. $\mathbf{x}_t=[x_{s1},x_{s2},...,x_{s(n/2)},x_{d1},...,x_{d(n/2)}]^T$. The total amount of feature vectors from all recordings of all the classes in the training set is denoted by T. The class and recording indices are omitted here since ICA does not utilize class information. The ICA demixing matrix **W** is applied on **X** producing the transformed observation space **O** = **WX**, which is of dimension m-by-T, where $m \le n$. The inequality is due to possible dimensionality reduction in the preprocessing step, which consists of a whitening transform.

The efficient FastICA algorithm was used for finding the ICA basis transformation [7]. It should be noted that the extra computational load caused by applying the ICA transformation occurs mainly in the off-line training phase. The test phase consists of computing the MFCC and ΔMFCC features in the usual way plus an additional multiplication with the m-by-n matrix **W** derived off-line using the training data.

3 CLASSIFICATION

3.1 The hidden Markov model

Hidden Markov models with a left-right topology are used to model the distribution of feature vectors from each instrument category, and the classification is made with the maximum-a-posteriori rule. A continuous density hidden Markov model (HMM) with N states consists of a set of parameters θ that comprises the N-by-N transition matrix, the initial state probabilities, and the parameters of the state densities. We use diagonal-covariance Gaussian-mixture state densities which are parameterized by the weights, means, and diagonal variances. The model parameters are estimated using a training set that consists of the recordings $\mathbf{O}=[\mathbf{O}^1,...,\mathbf{O}^R]$ and their associated class labels $L=(l^1,...,l^R)$. Specifically, $\mathbf{O}^r=[\mathbf{o}_1,...,\mathbf{o}_{T_r}]$ denotes the sequence of feature vectors measured from the recording r. The length of the observation sequence $\mathbf{O}^r$ is T_r. In this paper, each recording represents a single note played by an orchestral instrument, or a drum hit.

In our baseline system, the HMM parameters are iteratively optimized using the Baum-Welch re-estimation that finds a local maximum of the maximum likelihood (ML) objective function

$$F(\Theta)=\sum_{c=1}^{C}\sum_{r\in A_c}\log p(\mathbf{O}^r \mid c),$$

where Θ denotes the entire parameter set of all the classes $c\in\{1,...,C\}$, and A_c denotes the recordings from the class c. In the recognition phase, an unknown recording **Y** is classified using the maximum a posteriori rule:

$$\hat{c}=\arg\max_c p(\mathbf{Y}\mid c)$$

which is due to the Bayes' rule and assuming equal priors for all classes c. In this paper, the Viterbi-algorithm was used to approximate the above likelihoods.

3.2 Discriminative training

In the case that a statistical model fits poorly the data, training methods other than ML may lead into better-performing models. Discriminative training methods such as the maximum mutual information (MMI) aim at maximizing the ability to distinguish between the observation sequences generated by the model of the correct class and those generated by models of other classes [8]. The MMI objective function is given as

$$M(\Theta)=\log p(L\mid\mathbf{O})=\sum_{r=1}^{R}\log p(l^r\mid\mathbf{O}^r)$$
$$=\sum_{r=1}^{R}\left\{\log\left[p(l^r)\,p(\mathbf{O}^r\mid l^r)\right]-\log\sum_{c=1}^{C}p(c)\,p(\mathbf{O}^r\mid c)\right\}$$

where $p(l^r)$ and $p(c)$ are prior probabilities. Unfortunately, this requires rather complicated

optimization involving the entire model set even if observations from a single class were used.

In this paper, a recently-proposed discriminative training algorithm is used. The algorithm was proposed by Ben-Yishai and Burshtein, and is based on an approximation of the maximum mutual information [9]. Their *approximated maximum mutual information* (AMMI) criterion is:

$$J(\Theta)=\sum_{c=1}^{C}\left\{\sum_{r\in A_c}\log\left[p(c)p\left(\mathbf{O}^r|c\right)\right]-\lambda\sum_{r\in B_c}\log\left[p(c)p\left(\mathbf{O}^r|c\right)\right]\right\},$$

where B_c is the set of indices of training recordings that were *recognized* as *c*. B_c is obtained by maximum a posteriori classification performed on the training set, using initial models trained with the Baum-Welch algorithm. The "discrimination rate" is controlled using the parameter $0\le\lambda\le 1$.

The prior probabilities $p(c)$ do not affect the maximization of $J(\Theta)$, thus the maximization is equivalent to maximizing for all the classes $1\le c\le C$ the following objective functions:

$$J_c(\Theta)=\sum_{r\in A_c}\log p\left(\mathbf{O}^r|c\right)-\lambda\sum_{r\in B_c}\log p\left(\mathbf{O}^r|c\right).$$

Thus, the parameter set of each class can be estimated separately, which leads to a straightforward implementation. Ben-Yishai and Burshtein have derived the re-estimation equations for HMM parameters [9]. Due to space restrictions, we present only the re-estimation equation for the transition probability from state *i* to state *j*:

$$\bar{a}_{ij}=\frac{\sum_{r\in A_c}\sum_{t=1}^{T_r-1}\xi_t(i,j)-\lambda\sum_{r\in B_c}\sum_{t=1}^{T_r-1}\xi_t(i,j)}{\sum_{r\in A_c}\sum_{t=1}^{T_r-1}\gamma_t(i)-\lambda\sum_{r\in B_c}\sum_{t=1}^{T_r-1}\gamma_t(i)},$$

where $\xi_t(i,j)=p(q_t=i,q_{t+1}=j\,|\,\mathbf{O}^r,c)$ and $\gamma_t=\sum_{j=1}^{N}\xi_t(i,j)$. The state at time *t* is denoted by q_t, and the length of the observation sequence $\mathbf{O}^r$ is T_r. In a general form, for each parameter ν the re-estimation procedure is

$$\nu=\frac{N(\nu)-\lambda N_D(\nu)}{D(\nu)-\lambda D_D(\nu)}$$

where $N(\nu)$ and $D(\nu)$ are the accumulated statistics computed according to the set A_c, and $N_D(\nu)$ and $D_D(\nu)$ are the statistics computed according to the set B_c, obtained by recognition on the training set. Thus, in a typical situation the set B_c includes examples from the class *c* and some other confusing classes. This discriminative re-estimation can be iterated in a manner similar to the standard expectation-maximisation. We typically used 5 iterations, although using just one iteration seemed to be sufficient in many situations, since the recognition accuracy did not improve much after the first iteration.

4 VALIDATION EXPERIMENTS

4.1 Validation database

Our experimental setup aimed at testing the system's generalization ability across significant variations in recording setup and instrument instances. Samples from five different sources were used in the validation database. The sources were the McGill University Master Samples collection (MUMS) [10], the University of Iowa Electronic Music Studios website [11], IRCAM Studio Online [12], a Roland XP-30 synthesizer, and recordings arranged by Keith Martin at MIT Media Lab [2]. A total of 5895 samples of 27 Western orchestral instruments were included in the database, of which 4940 were included in the training set and 955 were tested. The division into training and test sets was done so that all the samples from a particular instrument instance in a certain recording session were either in the training or test set, i.e. the recognition was done across recordings and different instrument pieces. The recognition was performed at an intermediate level of abstraction using seven classes, which were *the brass, saxophones, single reed clarinets, double reed oboes, flutes, bowed strings,* and *plucked strings*. A random guesser would score 14% correct in these conditions. The drum database consisted of samples from 8 different synthesizer sound banks and the MUMS collection [10]. Samples of two sound banks were used in the training set (total of 1123 drum hits), and the samples of the seven remaining sources were used for testing (a total of 675). The five possible categories were *bass drum, cymbal, hi hat, snare,* and *tom-tom.*

4.2 Results

The Baum-Welch algorithm was used to train the baseline HMMs. The number of states (NS) and component densities per state (NC) was varied. Increasing the number of components in each state was obtained by gradually increasing the model order until the desired order NC was obtained by splitting the component with the largest weight. The state means and variances were initialized using a heuristic segmentation scheme, where each sound was segmented into as many adjacent segments as there were states in the model. The initial mean and variance for each state were estimated from the statistics accumulated from the different segments of all samples. During training, a straightforward form of regularization was applied by adding a small constant to the variance elements falling below a predetermined threshold.

Table 1 presents the results obtained using the baseline system using MFCC plus ΔMFCC features and HMMs trained using the Baum-Welch algorithm. In Table 2, the features have been ICA transformed; the HMM training is similar to the baseline. Table 3 shows the results using the MFCC plus ΔMFCC front-end, but using discriminative training of HMMs. In Table 4, both enhancements have been combined and the ICA transformed input is modelled with discriminatively trained HMMs. It can be observed

Table 1. Percentage correct in instrument identification, baseline system with MFCC plus ΔMFCC features and ML training.

% correct	NC=1	NC=2	NC=4	NC=6	NC=8
NS = 2	44	47	57	**60**	59
NS = 3	53	59	**60**	58	58
NS = 4	59	57	60	**62**	**62**
NS = 5	56	60	60	60	**62**

Table 2. Percentage correct in instrument identification, ICA-based transformation applied and ML training of HMMs.

% correct	NC=1	NC=2	NC=4	NC=6	NC=8
NS = 2	48	56	60	63	**66**
NS = 3	57	62	63	65	**67**
NS = 4	58	61	**66**	60	61
NS = 5	63	**66**	64	**66**	62

Table 3. Percentage correct in instrument identification, baseline features and discriminative training of HMMs.

% correct	NC=1	NC=2	NC=4	NC=6	NC=8
NS = 2	45	51	59	61	**62**
NS = 3	58	**63**	59	59	58
NS = 4	58	61	60	61	**64**
NS = 5	58	**62**	**62**	61	**62**

Table 4. ICA-based transformation applied and discriminative training of HMMs.

% correct	NC=1	NC=2	NC=4	NC=6	NC=8
NS = 2	51	57	61	65	**66**
NS = 3	57	64	64	66	**68**
NS = 4	60	60	**65**	61	61
NS = 5	65	**67**	63	65	62

that using the ICA transform gives an almost consistent improvement in recognition accuracy across the set of model orders tested. Using discriminative training improves the accuracy mainly with models having low number of components in state densities. This is understandable since low-order models give relatively low recognition accuracy in the training set, and there is not so much danger of over-fitting due to discriminative training as with higher order models. Different values of λ were tested, and the results are shown for λ =0.3.

Tables 5 and 6 show the results for the drum database using the baseline system and the ICA transformation. Here the improvement is not consistent across the different model orders evaluated, which may be partly due to the larger mismatch in training and testing conditions in this database, and the relatively smaller size of training data where examples from only two sound banks are included.

5 CONCLUSION

A system for the recognition of musical instrument samples was described. Applying an ICA-based transform of features gave an almost consistent improvement in recognition accuracy compared to the baseline. The

Table 5. Percentage correct in drum recognition, MFCC plus ΔMFCC features.

	NC = 1	**NC = 2**	**NC = 3**	**NC = 4**
NS = 2	79	79	80	78
NS = 3	76	77	79	81

Table 6. Percentage correct in drum recognition, ICA-based transformation applied.

	NC = 1	**NC = 2**	**NC = 3**	**NC = 4**
NS = 2	80	80	78	78
NS = 3	78	81	85	85

accuracy could be further improved by using discriminative training of the hidden Markov models. Future work will consider the extension of these methods for monophonic phrases.

ACKNOWLEDGMENT

We thank Assaf Ben-Yishai for his kind answers to our questions on discriminative training. The efforts of Iowa music studios, Ircam, and Keith Martin and Youngmoo Kim in making sound samples freely available are greatly appreciated.

REFERENCES

[1] J. C. Brown, "Feature dependence in the automatic identification of musical woodwind instruments". *J. Acoust. Soc. Am*., Vol. 109, No. 3, pp. 1064-1072, 2001.

[2] K. D. Martin, Sound-Source Recognition: *A Theory and Computational Model*. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, 1999. Available at http://sound.media.mit.edu/Papers/kdm-phdthesis.pdf.

[3] A. Eronen, "Comparison of features for musical instrument recognition". In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pp. 19-22, Oct. 2001.

[4] M. Casey, "Generalized Sound Classification and Similarity in MPEG-7". *Organized Sound*, 6:2, 2002.

[5] I. Potamitis, N. Fakotakis, G. Kokkinakis, "Independent component analysis applied to feature extraction for robust automatic speech recognition". *Electronics Letters*, Vol. 36, No. 23, Nov 2000.

[6] A. Kocsor, J. Csirik, "Fast Independent Component Analysis in Kernel Feature Spaces". *In Proc. SOFSEM 2001*, Springer-Verlag LNCS 2234, pp. 271-281, 2001.

[7] A. Hyvärinen. "Fast and Robust Fixed-Point Algorithms for Independent Component Analysis". *IEEE Transactions on Neural Networks* 10(3):626-634, 1999. Matlab software available at: http://www.cis.hut.fi/projects/ica/fastica/.

[8] L. R. Rabiner, B.-H. Juang. *Fundamentals of Speech Recognition*, PTR Prentice-Hall Inc., New Jersey, 1993.

[9] A. Ben-Yishai, D. Burshtein, "A Discriminative Training Algorithm for Hidden Markov Models". Submitted to *IEEE Transactions on Speech and Audio Processing*.

[10] F. Opolko, J. Wapnick, *McGill University Master Samples* (compact disk). McGill University, 1987.

[11] The University of Iowa Electronic Music Studios, website. http://theremin.music.uiowa.edu

[12] Ircam Studio Online, website. http://soleil.ircam.fr/

Publication 4

A. Eronen, V. Peltonen, J. Tuomi, A. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, J. Huopaniemi, "Audio-based context recognition", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 14, No. 1, pp. 321–329, January 2006.

Audio-Based Context Recognition

Antti J. Eronen, Vesa T. Peltonen, Juha T. Tuomi, Anssi P. Klapuri, Seppo Fagerlund, Timo Sorsa, Gaëtan Lorho, and Jyri Huopaniemi, *Member, IEEE*

***Abstract*—The aim of this paper is to investigate the feasibility of an audio-based context recognition system. Here, context recognition refers to the automatic classification of the context or an environment around a device. A system is developed and compared to the accuracy of human listeners in the same task. Particular emphasis is placed on the computational complexity of the methods, since the application is of particular interest in resource-constrained portable devices. Simplistic low-dimensional feature vectors are evaluated against more standard spectral features. Using discriminative training, competitive recognition accuracies are achieved with very low-order hidden Markov models (1–3 Gaussian components). Slight improvement in recognition accuracy is observed when linear data-driven feature transformations are applied to mel-cepstral features. The recognition rate of the system as a function of the test sequence length appears to converge only after about 30 to 60 s. Some degree of accuracy can be achieved even with less than 1-s test sequence lengths. The average reaction time of the human listeners was 14 s, i.e., somewhat smaller, but of the same order as that of the system. The average recognition accuracy of the system was 58% against 69%, obtained in the listening tests in recognizing between 24 everyday contexts. The accuracies in recognizing six high-level classes were 82% for the system and 88% for the subjects.**

***Index Terms*—Audio classification, context awareness, feature extraction, hidden Markov models (HMMs).**

I. Introduction

CONTEXT recognition is defined as the process of automatically determining the context around a device. Information about the context would enable wearable devices to provide better service to users' needs, e.g., by adjusting the mode of operation accordingly. A mobile phone can automatically go into an appropriate profile while in a meeting, refuse to receive calls, or a portable digital assistant can provide information customized to the location of the user [1].

Many sources of information for sensing the context are available, such as luminance, acceleration, or temperature. Audio provides a rich source of context-related information, and recognition of a context based on sound is possible for humans to some extent. Moreover, there already exist suitable sensors, i.e., microphones, in many portable devices.

In this paper, we consider context recognition using acoustic information only. Within this scope, a context denotes a location with different acoustic characteristics, such as a restaurant, marketplace, or a quiet room. Differences in the acoustic characteristics can be due either to the physical environment or the activity of humans and nature. We describe the collection of evaluation data representing the common everyday sound environment of urban people, allowing us to assess the feasibility of building context aware applications using audio. Using this data, a comprehensive evaluation is made of different features and classifiers. The main focus is on finding methods suitable for implementation on a mobile device. Therefore, we evaluate linear feature transforms and discriminative training to improve the accuracy obtained with very low-order HMMs.

An experiment was conducted to facilitate the direct comparison of the system's performance with that of human subjects. A forced-choice test with identical test samples and reference classes for the subjects and the system was used. We also made a qualitative test to assess the information on which the human subjects base their decision. To our knowledge, this study is the first attempt to present a comprehensive evaluation of a computer and human performance in audio-based context recognition. Some preliminary results on context recognition using audio have been described in [2], [3].

This paper is organized as follows. Section II reviews previous work. Section III presents the feature extraction algorithms used in this study. In Section IV, the classification methods are described. Section V presents an assessment of the computer system. In Section VI, a test on human perception of audio contexts is described. Finally, in Section VII, these results are compared to the performance of the system.

II. Previous Work

The research on context awareness is still at its early stages and very few applications have been constructed that make use of other context information than global positioning system (GPS) location [4]. One of the earliest prototypes of a context-aware system was the ParcTab developed at the Xerox Palo Alto Research Center [5]. The ParcTab featured, e.g., contextual information and commands, automatic contextual reconfiguration and context-triggered actions.

In many cases, the context-awareness functionality is build upon an array of different sensors sensing the context. In [6], the authors used accelerometers, photodiodes, temperature sensors, touch sensors, and microphones, from which simple low-level features were extracted. Another approach is to transform the

Manuscript received December 31, 2003; revised January 26, 2005. This work was supported by Nokia Research Center and TISE Graduate School. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Shoji Makino.

A. J. Eronen, T. Sorsa, G. Lorho, and J. Huopaniemi are with the Nokia Research Center, FIN-33721 Tampere, Finland (e-mail: antti.eronen@nokia.com; timo.sorsa@nokia.com; gaetan.lorho@nokia.com; jyri.huopaniemi@nokia.com).

V. T. Peltonen is with the Nokia Mobile Phones, FIN-33721 Tampere, Finland (e-mail: vesa.peltonen@nokia.com).

J. T. Tuomi and A. P. Klapuri are with the Institute of Signal Processing, Tampere University of Technology, FIN-33101 Tampere, Finland (e-mail: juha.tuomi@tut.fi; anssi.klapuri@tut.fi).

S. Fagerlund is with the Laboratory of Acoustics and Audio Signal Processing, Helsinki University of Technology, FIN-02015, Espoo, Finland (e-mail: seppo.fagerlund@hut.fi).

Digital Object Identifier 10.1109/TSA.2005.854103

raw input into a low-dimensional representation using principal component analysis (PCA) or independent component analysis (ICA) [7], [8].

In general, the process of context recognition is very similar regardless of the sensors or data sources used for the recognition. The feature vectors obtained from sensors are fed to classifiers that try to identify the context the particular feature vectors present. As classifiers, e.g., hidden Markov models (HMMs) [9], or a combination of a self-organizing map and a Markov chain, have been used [6].

Only few studies have attempted to classify contexts using acoustic information. Clarkson has classified seven contexts using spectral energies from the output of a filter bank and a HMM classifier [9]. In [10], Sawhney describes preliminary experiments with different features and classifiers in classifying between voice, traffic, subway, people, and others. The most successful system utilized frequency-band energies as features and a nearest-neighbor classifier.

El-Maleh *et al.* classified five environmental noise classes (a car, street, babble, factory, and bus) using line spectral features and a Gaussian classifier [11]. Couvreur *et al.* used HMMs to recognize five types of environmental noise events: car, truck, moped, aircraft, and train, using linear prediction cepstral coefficients as features and discrete HMMs [12]. The authors also described an informal listening test, which showed that, on the average, humans were inferior in classifying these categories compared to the system.

The features we are using are similar to those used in different audio information retrieval tasks [13]. Scheirer and Slaney described a speech/music discrimination system, which used a combination of several features [14]. More recent studies include that of Lu *et al.* [15] and Li *et al.* [16] who also included environmental noise as one of the categories. Zhang and Kuo [17] classified between harmonic environmental sound, nonharmonic environmental sound, environmental sound with music, pure music, song, speech with music, and pure speech.

Casey has used a front-end where log-spectral energies are transformed into a low-dimensional representation with singular-value decomposition and ICA [18]. The classifier uses single-Gaussian continuous-density HMMs with full covariance matrices trained with Bayesian maximum *a posteriori* (MAP) estimation. Casey's system was evaluated on a database consisting e.g., of musical instrument sounds, sound effects, and animal sounds.

To our knowledge, context recognition using audio has not been studied to this extent before. The results existing in the literature have used only a limited number of categories, often focusing into a certain noise type such as vehicle sounds. In this paper, we present results using comprehensive data measured from several everyday contexts. The most promising features presented in the literature are compared on this data. We propose a linear transformation of the concatenated cepstral and delta cepstral coefficients using PCA or ICA and show that this slightly improves the classification accuracy. Moreover, we demonstrate that compact diagonal-covariance Gaussian HMMs and discriminative training are an effective classifier for this task. To our knowledge, discriminatively trained HMMs have not been used for audio-based context recognition before.

III. Acoustic Measurements and Feature Extraction

A. Recording Procedure

To obtain a realistic estimate of the feasibility of building context-aware applications using audio input, we paid special attention to gathering a data set that would be representing of the everyday sound environment encountered by urban people. The recording procedure has been described in [19] and is summarized here. A total of 225 real-world recordings from a variety of different contexts were made using two different recording configurations. The first configuration has been developed by Zacharov and Koivuniemi [20]. It consists of a head-and-torso simulator with multiple microphones and is capable of storing multiple audio formats simultaneously. For the purpose of this study, we only utilized the binaural recordings (two channels) and stereo recordings (two channels). The microphones mounted in the ears of the dummy head enable a realistic binaural reproduction of an auditory scene. The stereo setup consisted of two omnidirectional microphones (AKG C460B), separated by a distance of one meter. This construction was attached to the dummy head. The acoustic material was recorded into a digital multitrack recorder in 16-bit and 48-kHz sampling rate format. A total of 55 recordings were made with this setup. The remaining measurements were made with an easily portable stereo setup using AKG C460B microphones.

The recording of spatial sound material was done for subjective evaluations. In computer simulations, we only used the left channel from the stereo setup. Table I shows the division of recordings into different categories.

B. Feature Extraction

A wide set of feature extractors was implemented for this study in order to evaluate the accuracy obtained with each, and to select a suitable feature set for the system.

All features are measured in short analysis frames. A typical analysis frame length in this study was 30 ms with 15-ms overlap. The hanning window function was used. The following features were evaluated in this study.

Zero-crossing rate (ZCR) is defined as the number of zero-voltage crossings within a frame.

Short-time average energy is the energy of a frame and is computed as the sum of squared amplitudes within a frame.

Mel-frequency cepstral coefficients (MFCC) are a perceptually motivated representation of the coarse shape of the spectrum [21]. We used 11 or 12 MFCC coefficients calculated from the outputs of a 40-channel filterbank.

Mel-frequency delta cepstral coefficients (ΔMFCC) are used to describe the dynamic properties of the cepstrum. We used a three-point linear fit to approximate the first time derivative of each cepstral coefficient.

Band-energy refers to the energies of subbands normalized with the total energy of the signal. We experimented with four and ten logarithmically-distributed subbands.

Spectral centroid represents the balancing point of the spectral power distribution.

Bandwidth is defined as the estimated bandwidth of the input signal [16].

TABLE I
STATISTICS OF THE AUDIO MEASUREMENTS

High level category	Context	Number of Recordings
Outdoors	Street	16
	Road	13
	Nature	12
	Construction	11
	Marketplace	1
	Fun Park	1
Vehicles	Car	27
	Bus	11
	Train	10
	Subway Train	6
Public / Social Places	Restaurant	13
	Cafeteria	10
	Pub	1
	Shop	13
	Lecture Pause	1
Offices / Meetings / Quiet Places	Office	12
	Lecture	12
	Meeting	4
	Library	11
Home	Living Room	2
	Kitchen	4
	Bathroom	6
	Music	2
Reverberant Places	Church	4
	Railway Station	11
	Subway Station	7
	Hall	4
Total		225

Spectral roll-off [16] measures the frequency below which a certain amount of spectral energy resides. It measures the "skewness" of the spectral shape.

Spectral flux (SF) is defined as the difference between the magnitude spectra of successive frames [14].

Linear prediction coefficients (LPCs) were extracted using the autocorrelation method [22, p. 103]. The number of LPC coefficients extracted was 12.

Linear prediction cepstral coefficients are obtained using a direct recursion from the LPC coefficients [22, p. 115]. The number of cepstral coefficients was 12 after discarding the zeroth coefficient.

All the features were mean and variance normalized using global estimates measured over the training data.

C. Feature Transforms

The main idea of linear data-driven feature-transformations is to project the original feature space into a space with a lower dimensionality and more feasible statistical properties, such as uncorrelatedness. In this work, three different techniques were used. The PCA finds a decorrelating transform [25, p. 115], ICA results in a base with statistical independence [25, p. 570], and the linear discriminant analysis (LDA) tries to maximize class separability [25, p. 120].

PCA projects the original data into a lower dimensional space such that the reconstruction error is as small as possible, measured as the mean-square error between the data vectors in the original space and in the projection space. Projection onto a lower dimensional space reduces the amount of parameters to be estimated in the classifier training stage, and uncorrelated features are efficiently modeled with diagonal-covariance Gaussians.

The goal of ICA is to find directions of minimum mutual information, i.e., to extract a set of statistically independent vectors from the training data. Here, the FastICA algorithm was used for finding the ICA basis transformation [23].

Himberg *et al.* have used PCA and ICA to project multidimensional sensor data from different contexts into a lower dimensional representation, but reported only qualitative results [4]. In speech recognition, the use of an ICA transformation has been reported to improve the recognition accuracy [24]. In the MPEG-7 generalized audio descriptors, ICA is proposed as an optional transformation for the spectrum basis obtained with singular value decomposition, and Casey's results have shown the success of this method on a wide variety of sounds [18]. Our approach is different from all these studies, since we perform ICA on concatenated MFCC and ΔMFCC features. Including the delta coefficients is a way to include information on temporal dependencies of features, which is ignored if the transform is applied on static coefficients only. In [18] and [24], delta coefficients were not considered.

The third feature transform technique tested here, LDA, differs from PCA and ICA by utilizing class information. The goal is to find basis vectors that maximize the ratio of between-class variance to within-class variance.

It should be noted that the extra computational load caused by applying any of these transformations occurs mainly in the off-line training phase. The test phase consists of computing the features in the usual way plus an additional multiplication once per analysis frame with the transform matrix derived off-line using the training data.

IV. CLASSIFICATION METHODS

A. K-Nearest Neighbors

The most straightforward classification method is nearest neighbor classification. The K-nearest-neighbors (K-NN) classifier performs a class vote among the k nearest training-data feature vectors to a point to be classified [25, p. 182]. In our implementation, the feature vectors were first decorrelated using PCA and the Euclidean distance metric was used in the transformed space. Averaging over 1-s-long segments was used to reduce the amount of calculations and required storage space.

B. HMM

1) Description of the Model: A HMM [22, pp. 321–386] is an effective parametric representation for a time-series of observations, such as feature vectors measured from natural sounds. In this work, HMMs are used for classification by training a HMM for each class, and by selecting the class with the largest *a posteriori* probability.

2) Model Initialization: We used the maximum-likelihood based Baum–Welch algorithm to train the "baseline" HMMs for each class separately. The number of states (NS) and the number of component densities per state (NC) was varied. The models were initialized with a single Gaussian at each state, and the component with the largest weight was then split until the

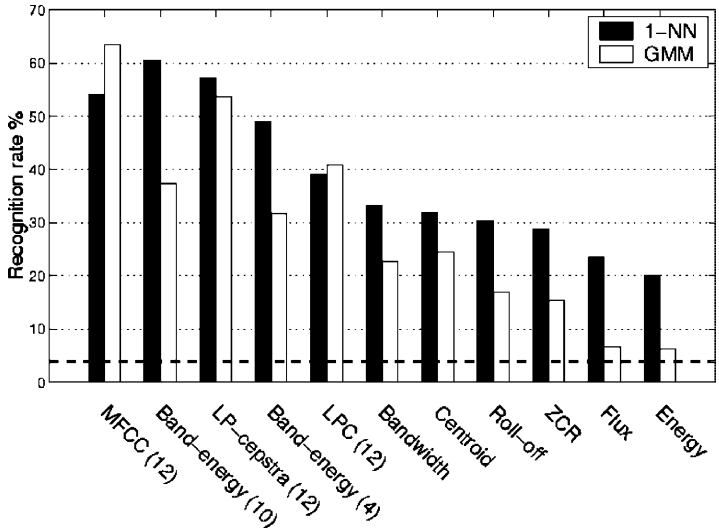


Fig. 1. Recognition accuracy obtained with different features using the GMM and 1-NN classifiers and 30 s of each test signal.

desired value of NC was obtained. Each component split was followed by 15 Baum–Welch iterations, or until the likelihood converged.

3) Discriminative Training of HMMs: In applications where computational resources are limited, we are forced to use models with as few Gaussians as possible, since their evaluation poses the computationally most demanding step in the recognition phase. In these cases the HMM is not able to fully represent the feature statistics and other approaches than maximum likelihood parameter estimation may lead into better recognition results. Discriminative training methods such as the maximum mutual information (MMI) aim at maximizing the ability to distinguish between the observation sequences generated by the model of the correct class and those generated by models of other classes [22, p. 363].

We used a discriminative training algorithm recently proposed by Ben–Yishai and Burshtein [26]. The algorithm is based on an approximation of the MMI. It starts from a "baseline" model set trained with the Baum–Welch algorithm, followed by an iterative discriminative training phase. At each discriminative training iteration, new statistics for the model parameters are accumulated not only from the observations of the correct class, but also from a set of confusing classes. The set of confusing classes is obtained by MAP classification performed on the training set. An interested reader should refer to [26] for more details of the algorithm.

V. Evaluation

A. Experimental Setup

Two training and testing setups were formed from the samples. Setup 1 consisted of 155 recordings of 24 contexts that were used for training and 70 recordings of 16 contexts were tested. Random division of recordings into the training and tests sets was done 100 times. The contexts selected into the test set had to have at least five recordings from different locations at different times. Setup 2 was used in the listening test and in the direct comparison, and had two nonoverlapping sets of 45 samples from 18 different contexts in the test set.

A higher level of abstraction may be sufficient for some applications. Hence, the recordings were also categorized into six high-level classes that are more general according to some common characteristics. These classes are: 1) outdoors, 2) vehicles, 3) public/social, 4) offices/meetings/quiet, 5) home, and 6) reverberant places. It should be noted that the allocation of individual contexts into high-level classes is ambiguous; many contexts can be associated with more than one high-level class.

B. Results

1) Comparison of Features: In the first experiment, we compared the accuracy obtained with different features. In this experiment, classification performance was evaluated using leave-one-out cross-validation on all the recorded data. The classifiers were trained with all recordings except the one that was left out for classification. In this way, the training data is maximally utilized but the system has never heard the test recording before. The overall recognition rate was calculated as the sample mean of the recognition rates of the individual contexts.

The recognition rates obtained at the context level using individual features with two different classifiers, the 1-NN and a one-state HMM (a GMM), are shown in Fig. 1. The test sequence duration was 30 s taken from the beginning of each test recording and the duration of each training recording was 160 s. The random guess rate for 24 classes is shown with the dashed line in Fig. 1. The 1-NN classifier performs on the average better than the GMM. This is indicative of complicated distributions of many features, which are not well modeled with a GMM with five diagonal-covariance Gaussians. The MFCC coefficients are well modeled with a GMM. With 12 MFCC features, we obtained a recognition accuracy of 63% using the GMM classifier, and with ten band-energy features the recognition accuracy was 61% using the 1-NN classifier.

2) Discriminative Training: The second experiment studied the HMM and the MFCC features in more detail. The MFCC coefficients were augmented with the delta coefficients. We trained models with different NSs and NCs, and varied the

TABLE II
RECOGNITION ACCURACY USING ONE-STATE HMMS WITH VARYING NUMBER OF COMPONENT DENSITIES

# Components	Baum-Welch	Discriminative
$NC = 1$	57 ± 4	60 ± 4
$NC = 2$	62 ± 4	63 ± 4
$NC = 3$	64 ± 4	65 ± 4
$NC = 4$	65 ± 4	66 ± 4
$NC = 5$	65 ± 4	66 ± 4

TABLE III
RECOGNITION ACCURACY (%) AND STANDARD DEVIATION USING HMMS WITH VARYING TOPOLOGIES AND NUMBER OF STATES

	Fully-Connected		Left-Right with Skips	
# States	Baum-Welch	Discriminative	Baum-Welch	Discriminative
$NS = 2$	60 ± 4	62 ± 4	- [a]	- [a]
$NS = 3$	61 ± 5	64 ± 5	62 ± 4	64 ± 5
$NS = 4$	63 ± 5	65 ± 5	63 ± 5	65 ± 5

[a]This topology is identical to the fully-connected with two states.

TABLE IV
RECOGNITION ACCURACY (%) AND STANDARD DEVIATION WHEN CONFUSIONS WITHIN THE SIX HIGHER LEVEL CLASSES ALLOWED

# States	Baum-Welch	Discriminative
$NS = 2$	75 ± 3	77 ± 3
$NS = 3$	77 ± 3	79 ± 3
$NS = 4$	77 ± 4	79 ± 4

model topology. The second aim was to compare the baseline maximum-likelihood training using the Baum–Welch algorithm and discriminative training. The division into training and test data was done according to Setup 1. The amount of training data used from each recording was 160 s. In order to obtain reliable accuracy estimates and to utilize the test data efficiently, the recognition was performed in adjacent 30-s windows with 25% overlap, and the final recognition result has been averaged over the different train/test divisions, recognition windows, recordings, and classes.

Tables II–IV show the results from this experiment. The baseline models were obtained after 15 Baum–Welch iterations. Three iterations of discriminative training were then applied on the models obtained from Baum–Welch re-estimation. Using an HMM with two or three states, or a one-state HMM with two or three component densities gives acceptable accuracies especially when discriminative training is used, taking into account the low computational demand of having to evaluate just a few diagonal covariance Gaussians.

3) Linear Feature Transforms: In the next experiment, we evaluated the use of the three linear feature transforms: PCA, ICA, and LDA. Table IV shows the recognition accuracies when the different transforms were applied on a feature vector consisting of concatenated MFCCs and their derivatives. On the average, applying the ICA or PCA transforms gives a slight improvement in recognition accuracy (Table V). In these experiments, we used a two-state HMM with one component density per state.

TABLE V
RECOGNITION ACCURACY USING LINEAR FEATURE TRANSFORMS

	No Transform	PCA	ICA	LDA
Context	61 ± 3	62 ± 3	62 ± 4	60 ± 4
Higher level	75 ± 3	76 ± 2	77 ± 3	76 ± 3

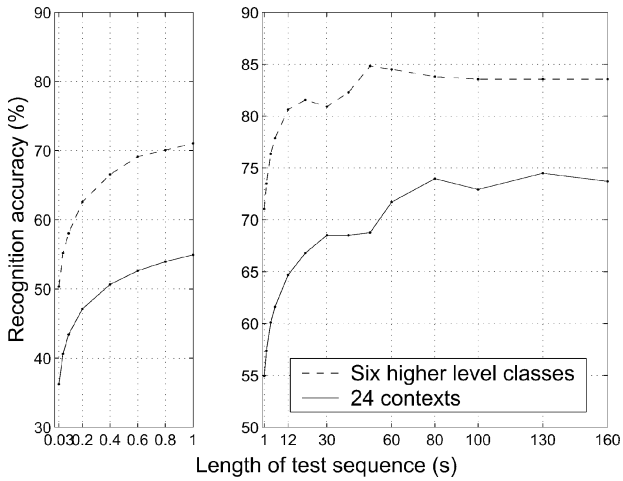


Fig. 2. Recognition accuracy as a function of test sequence length for the individual contexts and the six high-level classes. The left panel shows details of a test sequence length less than 1 s; the shortest length 0.03 corresponds to a single frame.

In [24], the authors reported improvements in speech recognition over the baseline using MFCC coefficients without a transform when these same transforms were applied either to the log-energy outputs of the MFCC filter bank, or the static MFCC coefficients. We made experiments also with these methods but improvement over the baseline was observed only when the concatenated MFCCs and deltas were transformed.

4) Effect of Test Sequence Length: In Fig. 2, the recognition rates obtained using the ICA transformed MFCC features and two-state HMMs are presented when the length of the test sequence was varied. The results for the six high-level classes have been derived from the results at the context level when confusions within the higher level categories are allowed.

As expected, increasing the length of test sequence improves the overall recognition rate. However, it takes rather long for the result to converge (around 60 s). With less than 20 s of test data, the recognition accuracy drops fast. Thus, this amount can be regarded as the lower limit for reliable recognition. The left panel shows the details with very short recognition sequence lengths ranging from just a single frame (30 ms) to 1 s. Even with these very short analysis segments some degree of accuracy can be obtained.

Presented \ Responded	Street	Road	Nature	Constr. site	Market place	Amusement park	Car	Bus	Train	Metro train	Restaurant	Cafe	Pub	Supermarket	Lecture pause	Office	Lecture	Meeting	Library	Living room	Kitchen	Bathroom	Music	Church	Railway station	Metro station	Hall
Street	**56**	35	3.7	1.9	1.9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Road	24	**70**	0	0	0	0	1.9	3.7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nature	0	1.9	**96**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Constr. site	3.7	1.9	0	**91**	0	1.9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.9
Car	3.7	0	0	0	0	0	**74**	7.4	7.4	5.6	0	0	0	0	0	0	0	0	0	1.9	0	0	0	0	0	0	0
Bus	1.9	1.9	0	1.9	1.9	1.9	3.7	**67**	5.6	11	0	0	3.7	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Train	0	0	0	0	0	0	9.3	11	**65**	1.9	1.9	0	0	0	0	3.7	0	0	3.7	0	0	0	0	0	0	1.9	1.9
Metro train	0	0	0	0	0	1.9	1.9	0	20	**72**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3.7	0
Restaurant	0	0	0	0	0	0	0	0	0	0	**70**	24	0	1.9	0	0	0	0	0	0	3.7	0	0	0	0	0	0
Cafe	1.9	0	0	0	1.9	3.7	0	0	0	0	17	**48**	5.6	3.7	3.7	1.9	0	0	0	3.7	3.7	0	0	0	1.9	1.9	1.9
Supermarket	0	0	0	0	0	1.9	0	0	0	0	1.9	9.3	1.9	**59**	0	0	0	0	0	3.7	0	0	3.7	0	3.7	1.9	13
Office	0	0	0	0	0	1.9	1.9	0	1.9	0	1.9	3.7	7.4	0	0	**59**	0	5.6	**11**	5.6	0	0	0	0	0	0	0
Lecture	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**91**	9.3	0	0	0	0	0	0	0	0	0
Meeting	0	0	0	0	0	0	0	0	0	0	1.9	3.7	1.9	0	9.3	17	1.9	**56**	0	1.9	7.4	0	0	0	0	0	0
Library	0	0	0	0	0	0	0	0	0	0	0	5.6	0	17	7.4	17	1.9	0	**35**	7.4	3.7	0	0	1.9	1.9	0	1.9
Bathroom	0	0	0	3.7	0	0	0	1.9	0	0	0	0	0	0	0	1.9	0	0	0	0	17	**76**	0	0	0	0	0
Railway station	0	0	0	1.9	3.7	3.7	0	0	3.7	1.9	0	0	0	1.9	0	0	0	0	0	0	0	0	0	0	**52**	1.9	30
Metro station	1.9	0	0	1.9	0	0	0	0	11	7.4	0	0	0	0	1.9	0	0	0	0	0	0	0	0	0	0	**69**	7.4

Fig. 3. Confusion matrix of the listening test experiment using stereo samples. The boxes indicate the high-level classes, which are (from left to right, top to bottom) outdoors, vehicles, public/social, offices/meetings/quiet, home, and reverberant.

VI. Human Perception of Audio Contexts

A. Setup of the Experiment

We also carried out an experiment on human recognition of audio contexts in order to obtain a performance baseline for the assessment of the system. This experiment was organized in three listening tests.

1) Stimuli, Reproduction System, and Listening Conditions: The stimuli for the listening tests were the recordings from the Setup 2 as described in Section V-A. All stimuli employed in this experiment were 1-min-long samples and were defined using two levels of categorization: context and high-level context.

All tests were performed in an ITU-R BS.1116-1 compliant listening room [27]. Audio samples were reproduced at a natural sound level over a stereophonic setup using Genelec 1031A loudspeakers placed at $\pm 30^\circ$ in front of the listener. The test design and administration were performed using the Presentation software [28]. This system allows very accurate monitoring of the reaction time between sample replay and subject responses.

2) Description of the Three Listening Tests: The focus of the main test was in studying the accuracy and reaction time of humans in audio context recognition. The second test compared the human ability in recognition with three different sound configurations, namely, the monophonic, stereophonic, and binaural reproduction techniques, in an assumed order of increasing degree of spaciousness. A subset of 18 samples from nine different contexts was selected for each configuration in this part of the experiment. For the binaural samples, crosstalk cancellation filters were designed based on the MIT KEMAR HRTF measurements [29] in order to obtain appropriate reproduction of the signal over loudspeakers (i.e., a binaural to transaural conversion).

The aim of the third test was to obtain a qualitative description of the recognition of auditory scenes. Subjects were asked to listen to nine samples and rate the information they used in the recognition process. After each stimulus, listeners filled in a form in which they were asked to evaluate and rate on a six-point discrete scale, how important different cues were in recognition (0 accounted for a cue not used and 5 for a cue considered very important).

In the three tests, subjects were instructed to try to recognize the context as fast as possible. A list of possible contexts was given to the test subjects. The list included also contexts not presented during the test. Recognition time was measured from the starting time of the stimulus presentation to the first keyboard press, after which the subject could select the context recognized by an additional keyboard press.

Eighteen subjects participated in the test, which was designed for two groups, each including the same number of stimuli and identical contexts. This permitted the use of more samples from the database, still keeping the total duration of the test within 1 h. The listening test started with a training session including nine samples not included in the actual test to familiarize the subjects with the user interface and the test setup.

B. Results of the Listening Test

Two measures were analyzed from this listening test, the recognition rate and the reaction time for each stimulus. Statistical methods employed were different due to the different nature of the two measures. First, recognition rate was analyzed as a set of right or wrong answers using a nonparametric statistical procedure, i.e., the Friedman and Kruskal–Wallis tests. For the reaction time, the statistical analysis was performed with a classical parametric statistical procedure (ANOVA), after discarding data considered as outliers.

1) Stereo Test: Rate was calculated for both context and high-level context recognition. As a result, the average recognition rate was 69% for contexts and 88% for the high-level contexts. Fig. 3. presents the confusion matrix for this experiment averaged over all listeners (differences between the two groups are not significant). Context and high-level context with

Presented \ Responded	Street	Road	Nature	Constr. site	Market place	Amusement park	Car	Bus	Train	Metro train	Restaurant	Cafe	Pub	Supermarket	Lecture pause	Office	Lecture	Meeting	Library	Living room	Kitchen	Bathroom	Music	Church	Railway station	Metro station	Hall
Street	**0**	0	0	75	0	0	0	0	0	0	0	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Road	0	**100**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nature	0	0	**83**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	17	0	0	0	0	0	0	0	0
Constr. site	0	0	0	**25**	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	0	25	0	0	0	0	0
Car	0	0	0	0	0	0	**100**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bus	0	0	0	0	0	0	17	**67**	0	0	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0
Train	0	0	0	0	0	0	0	0	**50**	0	0	25	0	0	0	25	0	0	0	0	0	0	0	0	0	0	0
Metro train	0	0	0	0	0	17	0	0	0	**50**	17	0	13	3.7	0	0	0	0	0	0	0	0	0	0	0	0	0
Restaurant	0	0	0	0	0	0	0	0	0	0	**25**	50	0	25	0	0	0	0	0	0	0	0	0	0	0	0	0
Cafe	0	0	0	0	0	0	0	0	0	17	46	**16**	4.7	17	0	0	0	0	0	0	0	0	0	0	0	0	0
Supermarket	0	0	0	0	0	0	0	0	0	0	3	6	0	**66**	0	25	0	0	0	0	0	0	0	0	0	0	0
Office	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**75**	0	0	25	0	0	0	0	0	0	0	0
Lecture	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**100**	0	0	0	0	0	0	0	0	0	0
Meeting	0	0	0	0	0	0	0	0	0	0	0	0	0	0	17	33	50	**0**	0	0	0	0	0	0	0	0	0
Library	0	0	0	0	0	0	0	0	0	0	0	0	0	17	0	0	0	15	**68**	0	0	0	0	0	0	0	0
Bathroom	0.3	0	0.3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	2	**81**	0	0	0	0	0
Railway station	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**75**	25	0
Metro station	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	25	0	0	**74**	0

Fig. 4. Confusion matrix when the system was tested on the samples from the listening test. Compare this to Fig. 3. The boxes indicate the high-level classes, which are (from left to right, top to bottom): Outdoors, vehicles, public/social, offices/meetings/quiet, home, and reverberant.

TABLE VI
RECOGNITION ACCURACY (%) FOR THE DIFFERENT PRESENTATION TECHNIQUES

	Mono	Stereo	Binaural	Average
Context	63	70	62	66
Higher-level	86	89	90	88

TABLE VII
CUES USED FOR AUDIO CONTEXT RECOGNITION

	Human activity	Spatial information	Prominent event	Continuous voice	Vehicle noise	Nature sounds
Cues used	67%	67%	64%	47%	32%	8%
Importance of the cue	2.55	1.88	2.50	1.89	1.77	2.26

the highest recognition rate were respectively *nature* (96%) and *outdoors* (97%), whereas those with the lowest rate were *library* (35%) and *office/other quiet places* (76%). Reaction time was also compared for the 18 contexts. Overall, the average reaction time was 13 s, ranging from 5 s (*nature*) to 21 s (*library*).

2) Mono/Stereo/Binaural Test: In the analysis of the second test, recognition rates were compared for monophonic, stereophonic, and binaural presentations. The average rate for context recognition with the three presentation techniques is shown in Table VI. The recognition rate averaged over the three techniques was 66% for context and it increased to 88% for high-level contexts. Differences in recognition accuracy can be observed between the different presentation techniques, especially with the stereo configuration in the case of context recognition, but this is not statistically significant overall. An average recognition time of 14 s was found for all stimuli. Comparing now the three presentation techniques, a significant difference was found with lower average recognition time for the stereo and binaural presentation (13 s) than the mono one (15 s).

3) Qualitative Test: In the last test, data on the qualitative assessment of recognition cues was collected and analyzed. The two measures computed from the questionnaire were a percentage of specific cues used in recognition (i.e., cue not used for a 0 rating and cue used otherwise) and its importance for the recognition process (i.e., an average of rates over subjects), as shown in Table VII. As a result, it was found that human activity and spatial information cues are most often used (67% of cases), with a lower importance for spatial information, however (1.88 rating against 2.55 for human activity). Prominent events were also mentioned as an important cue for recognition with a rate of 2.50.

C. Conclusion of the Subjective Test

This listening test showed that humans are able to recognize contexts in 69% of cases. The recognition rate increases to 88%, when considering high-level categorization of contexts only. Recognition time was 13 s on average. It should be noted, however, that reaction time for high-level context detection alone would probably be significantly faster. Indeed, some of the subjects reported that they could exclude most of the contexts fast, but the final decision between specific contexts from the same high-level context class took more time. Differences between the different reproduction techniques were also found, but these were not statistically significant. The presentation technique was only found to be significant for the reaction time.

D. Performance Comparison Between the System and Human Listeners

A direct comparison between the system and the human ability was made using exactly the same test samples and reference classes as in the listening test. Figs. 3 and 4 show the

averaged confusion matrices for the subjects and the system on this test setup, respectively. The boxes indicate the six high-level categories. The amount of test data given to the system was 30 s, since the human subjects did not usually listen through the whole 60 s. The averaged recognition accuracies of the computer system are 58% and 82% against the accuracies 69% and 88% obtained in the listening test for contexts and high-level classes, respectively.

VII. Conclusion

Building context aware applications using audio is feasible, especially when high-level contexts are concerned. In comparison with the human ability, the proposed system performs rather well (58% versus 69% for contexts and 82% versus 88% for high-level classes for the system and humans, respectively). Both the system and humans tend to make similar confusions mainly within the high-level categories.

Computationally efficient recognition methods were evaluated. Quite reliable recognition can be achieved using only a four-dimensional feature vector that represents subband energies, and even simplistic one-dimensional features achieve recognition accuracy significantly beyond chance rate. Discriminative training leads to slightly but consistently better recognition accuracies particularly for low-order HMM models. Slight increase in recognition accuracy can also be obtained by using PCA or ICA transformation of the mel-cepstral features.

The recognition rate as a function of the test sequence length appears to converge only after about 30 to 60 s. Some degree of accuracy can be achieved even in analysis frames below 1 s. The average reaction time of human listeners was 14 s, i.e., somewhat smaller but of the same order as that of the system.

References

[1] J. Mäntyjärvi, P. Huuskonen, and J. Himberg, "Collaborative context determination to support mobile terminal applications," *IEEE Wireless Commun.*, no. 5, pp. 39–45, Oct. 2002.

[2] V. Peltonen, J. Tuomi, A. Klapuri, J. Huopaniemi, and T. Sorsa, "Computational auditory scene recognition," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, May 2002, pp. 1941–1944.

[3] A. Eronen, J. Tuomi, A. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi, "Audio-based context awareness – Acoustic modeling and perceptual evaluation," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 5, Apr. 2002, pp. 529–532.

[4] G. Chen and D. Kotz, "A survey of context-aware mobile computing research," Dept. Comp. Sci., Dartmouth College, Hanover, NH, Tech. Rep. TR2000-381, 2000.

[5] B. N. Schilit, N. Adams, R. Gold, M. Tso, and R. Want, "The PARCTAB mobile computing system," in *Proc. IEEE 4th Workshop on Workstation Operating Systems*, Oct. 1993, pp. 34–39.

[6] K. Van Laerhoven, K. Aidoo, and S. Lowette, "Real-time analysis of data from many sensors with neural networks," in *Proc. 5th Int. Symp. Wearable Computers*, 2001, pp. 115–123.

[7] J. Himberg, J. Mäntyjärvi, and P. Korpipää, "Using PCA and ICA for exploratory data analysis in situation awareness," in *Proc. IEEE Int. Conf. Multisensor Fusion and Integration for Intelligent Systems*, Sep. 2001, pp. 127–131.

[8] F. M. Salam and G. Erten, "Sensor fusion by principal and independent component decomposition using neural networks," in *Proc. IEEE Int. Conf. Multisensor Fusion and Integration for Intelligent Systems*, Aug. 1999, pp. 127–131.

[9] B. Clarkson and A. Pentland, "Unsupervised clustering of ambulatory audio and video," Perceptual Computing Group, MIT Media Lab, Cambridge, MA, Tech. Rep. 471.

[10] N. Sawhney, "Situational awareness from environmental sounds," Project Rep., Speech Interface Group, MIT Media Lab, Cambridge, MA, 1997.

[11] K. El-Maleh, A. Samouelian, and P. Kabal, "Frame level noise classification in mobile environments," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 1, Mar. 1999, pp. 237–240.

[12] C. Couvreur, V. Fontaine, P. Gaunard, and C. G. Mubikangiey, "Automatic classification of environmental noise events by hidden Markov models," *Appl. Acoust.*, vol. 54, no. 3, pp. 187–206, 1998.

[13] J. Foote, "An overview of audio information retrieval," *Multimedia Syst.*, vol. 7, pp. 2–10, 1999.

[14] E. Scheirer and M. Slaney, "Construction and evaluation of a robust multifeature speech/music discriminator," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 2, Apr. 1997, pp. 1331–1334.

[15] L. Lu, H.-J. Zhang, and H. Jiang, "Content analysis for audio classification and segmentation," *IEEE Trans. Speech Audio Process.*, vol. 10, no. 5, pp. 504–516, Sep. 2002.

[16] D. Li, I. K. Sethi, N. Dimitrova, and T. McGee, "Classification of general audio data for content-based retrieval," *Pattern Recognit. Lett.*, no. 22, pp. 533–544, 2001.

[17] T. Zhang and C.-C. J. Kuo, *Content-Based Audio Classification and Retrieval for Audiovisual Data Parsing*. Norwell, MA: Kluwer, 2000.

[18] M. Casey, "Generalized sound classification and similarity in MPEG-7," *Org. Sound*, vol. 6, no. 2, 2002.

[19] V. Peltonen, "Computational auditory scene recognition," M.S. thesis, Dept. Inf. Technol., Tampere Univ. Technol., Tampere, Finland, 2001.

[20] N. Zacharov and K. Koivuniemi, "Unraveling the perception of spatial sound reproduction: Techniques and experimental design," presented at the Audio Eng. Soc. 19th Int. Conf. Surround Sound, Techniques, Technology and Perception, Jun. 2001.

[21] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 28, no. 4, pp. 357–366, Aug. 1980.

[22] L. R. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Upper Saddle River, NJ: Prentice-Hall, 1993.

[23] A. Hyvärinen, "Fast and robust fixed-point algorithms for independent component analysis," *IEEE Trans. Neural Netw.*, vol. 10, no. 3, pp. 626–634, May 1999.

[24] I. Potamitis, N. Fakotakis, and G. Kokkinakis, "Independent component analysis applied to feature extraction for robust automatic speech recognition," *Electron. Lett.*, vol. 36, no. 23, Nov. 2000.

[25] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: Wiley, 2001.

[26] A. Ben-Yishai and D. Burshtein, "A discriminative training algorithm for hidden Markov models," *IEEE Trans. Speech Audio Process.*, vol. 12, no. 3, pp. 204–217, May 2004.

[27] "Methods for the subjective assessment of small impairments in audio systems including multichannel sound systems," Int. Telecommun. Union Radiocommun. Assembly, ITU-R, Recommendation BS.1116-1, 1997.

[28] "Presentation" [Software]. Neurobehavioral Systems. [Online]. Available: http://www.neurobehavioralsystems.com/software/presentation/

[29] W. G. Gardner and K. D. Martin, "HRTF measurements of a KEMAR dummy-head microphone," Perceptual Computing Group, MIT Media Lab, Cambridge, MA, Tech. Rep. 280, 1994.

Antti J. Eronen was born in Ilomantsi, Finland, in 1977. He received the M.S. degree in information technology from the Tampere University of Technology (TUT), Tampere, Finland, in 2001. He is currently pursuing the Ph.D. degree.

From 1998 to 2003, he was with the Institute of Signal Processing, TUT. In 2003, he joined Nokia Research Center, Tampere. His research interests include content recognition, analysis, and synthesis of audio and music.

Vesa T. Peltonen was born in Virolahti, Finland, in 1974. He received the M.Sc. degree in information technology from the Tampere University of Technology (TUT), Tampere, Finland, in August 2001.

From 2000 to 2002, he was a Researcher with the Digital Media Institute, TUT. Since 2002, he has been with Nokia Mobile Phones, Tampere.

Juha T. Tuomi was born in Seinäjoki, Finland, in 1979. He is currently pursuing the M.S. degree in audio signal processing at the Tampere University of Technology (TUT), Tampere, Finland, with a thesis on auditory context tracking.

He joined the Institute of Signal Processing, TUT, in 2001, and has since been working on auditory context awareness. His principal focus is robust auditory context transition detection. His other research interests include mobile context awareness and the human perception of auditory contexts.

Anssi P. Klapuri was born in Kälviä, Finland, in 1973. He received the M.Sc. degree in information technology and the Ph.D. degree from the Tampere University of Technology (TUT), Tampere, Finland, in June 1998 and April 2004, respectively.

He has been with the Institute of Signal Processing, TUT, since 1996. His research interests include audio signal processing and, particularly, automatic transcription of music.

Seppo Fagerlund was born in Pori, Finland, in 1978. He is currently pursuing the M.S. degree at the Helsinki University of Technology (HUT), Espoo, Finland, with a thesis on automatic recognition of bird species by their sounds.

In 2002, he was a Research Assistant at Nokia Research Center, Tampere, Finland. In 2004, he became a Research Assistant at the Laboratory of Acoustics and Audio Signal Processing, HUT. His research interests include signal processing of bioacoustic signals and pattern recognition algorithms.

Timo Sorsa was born in Helsinki, Finland, in 1973. He received the M.Sc. degree in electrical engineering from the Helsinki University of Technology, Espoo, Finland, in 2000.

In 1999, he joined the Nokia Research Center, Helsinki, where he is currently with the Multimedia Technologies Laboratory. His current research interests include perceptual audio quality, audio content analysis, and audio signal processing.

Gaëtan Lorho was born in Vannes, France, in 1972. He received the M.S. degree in fundamental physics from the University of Paris VII, Paris, France, in 1996, and the M.S. degree in sound and vibration studies from the Institute of Sound and Vibration Research, University of Southampton, Southampton, U.K., in 1998.

Since 1999, he has been a Research Engineer at the Nokia Research Center, Helsinki, Finland. His main research interests are in the subjective evaluation of audio quality, spatial sound reproduction, and audio user interfaces.

Jyri Huopaniemi (M'99) was born in Helsinki, Finland, in 1968. He received the M.Sc., Lic.Tech., and D.Sc. (Tech.) degrees in electrical and communications engineering from the Helsinki University of Technology (HUT), Espoo, Finland, in 1995, 1997, and 1999, respectively. His doctoral thesis was on the topic of virtual acoustics and 3-D audio.

He was a Research Scientist at the Laboratory of Acoustics and Audio Signal Processing, HUT, from 1993 to 1998. In 1998, he was a Visiting Scholar at the Center for Computer Research in Music and Acoustics (CCRMA), Stanford University, Stanford, CA. Since 1998, he has been with Nokia Research Center, Helsinki, where his current position is Head of Mobile Applications Research. His professional interests include multimedia, software platforms, virtual audio-visual environments, digital audio signal processing, room and musical acoustics, and audio content analysis and processing. He is author or coauthor of over 55 technical papers published in international journals and conferences, and he has been actively involved in MPEG and Java standardization work.

Dr. Huopaniemi is a member of the AES and the Acoustical Society of Finland.

Publication 5

A. Klapuri, A. Eronen, J. Astola, "Analysis of the meter of acoustic musical signals", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 14, No. 1, pp. 342–355, January 2006.

Analysis of the Meter of Acoustic Musical Signals

Anssi P. Klapuri, Antti J. Eronen, and Jaakko T. Astola, *Fellow, IEEE*

***Abstract*—A method is decribed which analyzes the basic pattern of beats in a piece of music, the musical meter. The analysis is performed jointly at three different time scales: at the temporally atomic tatum pulse level, at the tactus pulse level which corresponds to the tempo of a piece, and at the musical measure level. Acoustic signals from arbitrary musical genres are considered. For the initial time-frequency analysis, a new technique is proposed which measures the degree of musical accent as a function of time at four different frequency ranges. This is followed by a bank of comb filter resonators which extracts features for estimating the periods and phases of the three pulses. The features are processed by a probabilistic model which represents primitive musical knowledge and uses the low-level observations to perform joint estimation of the tatum, tactus, and measure pulses. The model takes into account the temporal dependencies between successive estimates and enables both causal and noncausal analysis. The method is validated using a manually annotated database of 474 music signals from various genres. The method works robustly for different types of music and improves over two state-of-the-art reference methods in simulations.**

***Index Terms*—Acoustic signal analysis, music, musical meter analysis, music transcription.**

I. Introduction

METER analysis, here also called *rhythmic parsing*, is an essential part of understanding music signals and an innate cognitive ability of humans even without musical education. Perceiving the meter can be characterized as a process of detecting moments of musical stress (accents) in an acoustic signal and filtering them so that underlying periodicities are discovered [1], [2]. For example, tapping a foot to music indicates that the listener has abstracted metrical information about music and is able to predict when the next beat will occur.

Musical meter is a hierarchical structure, consisting of pulse sensations at different levels (time scales). Here, three metrical levels are considered. The most prominent level is the *tactus*, often referred to as the foot tapping rate or the beat. Following the terminology of [1], we use the word *beat* to refer to the individual elements that make up a pulse. A musical meter can be illustrated as in Fig. 1, where the dots denote beats and each sequence of dots corresponds to a particular pulse level. By the *period* of a pulse we mean the time duration between successive beats and by *phase* the time when a beat occurs with respect to the beginning of the piece. The *tatum* pulse has its name

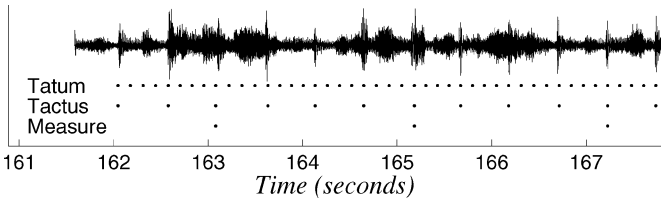


Fig. 1. Music signal with three metrical levels illustrated.

stemming from "temporal atom" [3]. The period of this pulse corresponds to the shortest durational values in music that are still more than incidentally encountered. The other durational values, with few exceptions, are integer multiples of the tatum period and the onsets of musical events occur approximately at a tatum beat. The *musical measure* pulse is typically related to the harmonic change rate or to the length of a rhythmic pattern. Although sometimes ambiguous, these three metrical levels are relatively well-defined and span the metrical hierarchy at the aurally most important levels. The *tempo* of a piece is defined as the rate of the tactus pulse. In order that a meter would make sense musically, the pulse periods must be slowly varying and, moreover, each beat at the larger levels must coincide with a beat at all the smaller levels.

The concept *phenomenal accent* is important for meter analysis. Phenomenal accents are events that give emphasis to a moment in music. Among these are the beginnings of all discrete sound events, especially the onsets of long pitched events, sudden changes in loudness or timbre, and harmonic changes. Lerdahl and Jackendoff define the role of phenomenal accents in meter perception compactly by saying that "the moments of musical stress in the raw signal serve as cues from which the listener attempts to extrapolate a regular pattern" [1, p. 17].

Automatic rhythmic parsing has several applications. A metrical structure facilitates cut-and-paste operations and editing of music signals. It enables synchronization with light effects, video, or electronic instruments, such as a drum machine. In a disc jockey application, metrical information can be used to mark the boundaries of a rhythmic loop or to synchronize two audio tracks. Provided that a time-stretching algorithm is available, rhythmic modifications can be made to audio signals [4]. Rhythmic parsing for musical instrument digital interface (MIDI)[1] data is required for *time quantization*, an indispensable subtask of score typesetting from keyboard input [5]. The particular motivation for the present work is to utilize metrical information in further signal analysis and in music transcription [6]–[8].

Manuscript received January 15, 2004; revised September 28, 2004. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Michael Davies.

A. P. Klapuri and J. T. Astola are with the Institute of Signal Processing, Tampere University of Technology, FIN-33720 Tampere, Finland (e-mail: Anssi.Klapuri@tut.fi; Jaakko.Astola@tut.fi).

A. J. Eronen is with the Nokia Research Center, Audio-Visual Systems Laboratory, FIN-33721 Tampere, Finland (e-mail: Antti.Eronen@nokia.com).

Digital Object Identifier 10.1109/TSA.2005.854090

[1]A standard interface for exchanging performance data and parameters between electronic musical devices.

A. Previous Work

The work on automatic meter analysis originated from algorithmic models that attempted to explain how a human listener arrives at a particular metrical interpretation of a piece. An extensive analysis of the early models has been given by Lee in [9] and later augmented by Desain and Honing in [10]. In brief, the early models performed meter analysis for symbolic data (impulse patterns) and can be seen as being based on a *set of rules* that were used to define what makes a musical accent and to infer the most natural meter.

More recently, Rosenthal proposed a system to emulate the human rhythm perception for piano performances, presented as MIDI files [11]. Parncutt developed a detailed algorithmic model of meter perception based on systematic listening tests [12]. Brown analyzed the meter of musical scores by processing the onset times and durations of note events using the autocorrelation function [13]. Large and Kolen used adaptive oscillators which adjust their period and phase to an incoming pattern of impulses, located at the onsets of musical events [14].

Temperley and Sleator [15] proposed a meter analysis algorithm for arbitrary MIDI files by implementing the preference rules that were described in verbal terms by Lerdahl and Jackendoff in [1]. Dixon proposed a rule-based system to track the tactus pulse of expressive MIDI performances and introduced a simple onset detector to make the system applicable for audio signals [16]. The source codes of both Temperley's and Dixon's systems are publicly available for testing.

Cemgil and Kappen developed a probabilistic generative model for the timing deviations in expressive musical performances [5]. Then, the authors used Monte Carlo methods to infer a hidden continuous tempo variable and quantized ideal note onset times from observed noisy onset times in a MIDI file. A similar Bayesian model was independently proposed by Raphael [17].

Goto and Muraoka were the first to achieve a reasonable meter analysis accuracy for audio signals [18], [19]. Their system operated in real time and was based on an architecture where multiple agents tracked competing meter hypotheses. Beat positions at the larger levels were inferred by detecting certain drum sounds [18] or chord changes [19].

Scheirer proposed an approach to tactus tracking where no discrete onsets or sound events are detected as a middle-step, but periodicity analysis is performed directly on the half-wave rectified (HWR) differentials of subband power envelopes [20]. The source code of Scheirer's system is publicly available. Sethares and Staley took a similar approach, but used a periodicity transform for periodicity analysis instead of a bank of comb filters [21]. Laroche proposed a noncausal algorithm where spectral change was measured as a function of time, the resulting signal was correlated with impulse trains of different periods, and dynamic programming was used to find a continuous time-varying tactus pulse [22].

Hainsworth and Macleod [23] developed a method which is loosely related to that of Cemgil *et al.* [5]. They extracted discrete onsets from an audio signal and then used particle filters to associate the onsets to a time-varying tempo process and to find the locations of the beats. Gouyon *et al.* proposed a system for detecting the tatum pulse in percussive audio tracks of constant tempo [24].

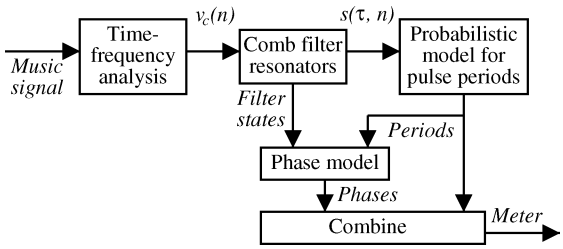


Fig. 2. Overview of the meter estimation method. The two intermediate data representations are bandwise accent signals $v_c(n)$ and metrical pulse saliences (weights) $s(\tau, n)$.

In summary, most of the earlier work on meter analysis has concentrated on symbolic (MIDI) data and typically analyzed the tactus pulse only. Some of the systems [5], [14], [16], [17] can be immediately extended to process audio signals by employing an onset detector which extracts the beginnings of discrete acoustic events from an audio signal. Indeed, the authors of [16] and [17] have introduced an onset detector themselves. Elsewhere, onset detection methods have been proposed that are based on using subband energies [25], an auditory model [26], support vector machines [27], independent component analysis [28], or a complex-domain distance measure [29]. However, if a rhythmic parser has been originally developed for symbolic data, the extended system is usually not robust to diverse acoustic material (e.g., classical versus rock music) and cannot fully utilize the acoustic cues that indicate phenomenal accents in music signals.

There are a few basic problems that need to be addressed in a successful meter analysis system. First, the degree of musical accentuation as a function of time has to be measured. Some systems do this in a continuous manner [20], [21] whereas others extract discrete onsets from an audio signal [18], [22], [24]. Second, the periods and phases of the underlying metrical pulses have to be estimated. The methods which detect discrete events as a middle-step have often used inter-onset-interval histograms for estimating the periods [16], [18], [19], [24]. Third, a system has to choose the metrical level which corresponds to the tactus or some other specially designated pulse level. This may take place implicitly, or by using a prior distribution for pulse periods [12] or by rhythmic pattern matching [18].

B. Proposed Method

The aim of this paper is to describe a method which analyzes the meter of acoustic musical signals at the tactus, tatum, and measure pulse levels. The target signals are not limited to any particular music type but all the main Western genres, including classical music, are represented in the validation database.

An overview of the method is shown in Fig. 2. For the time-frequency analysis part, a technique is proposed which aims at measuring the degree of accentuation in a music signal. The technique is robust to diverse acoustic material and can be loosely seen as a synthesis and generalization of two earlier state-of-the-art methods [18] and [20]. Feature extraction for estimating the pulse periods and phases is performed using comb filter resonators very similar to those used by Scheirer in

[20]. This is followed by a probabilistic model where the period-lengths of the tactus, tatum, and measure pulses are jointly estimated and temporal continuity of the estimates is modeled. At each time instant, the periods of the pulses are estimated first and act as inputs to the phase model. The probabilistic models encode prior musical knowledge and lead to a more reliable and temporally stable meter tracking. Both causal and noncausal algorithms are presented.

This paper is organized as follows. Section II will describe the different elements of the system shown in Fig. 2. Section III will present experimental results and compare the proposed method with two reference methods. The main conclusions will be summarized in Section IV.

II. Meter Analysis Model

This section will describe the different parts of the meter analysis method illustrated in Fig. 2. Section II-A will describe the time-frequency analysis part. In Section II-B, the comb filter resonators will be introduced. Sections II-C and II-D will describe the probabilistic models which are used to estimate the periods and phases of the three pulse levels.

A. Calculation of Bandwise Accent Signals

All the phenomenal accent types mentioned in the introduction can be observed in the time-frequency representation of a signal. Although an analysis using a model of the human auditory system might seem theoretically advantageous (since meter is basically a cognitive phenomenon), we did not manage to obtain a performance advantage using a model similar to [26] and [30]. Also, the computational complexity of such models makes them rather impractical.

In a time-frequency plane representation, some data reduction must take place to discard information which is irrelevant for meter analysis. A big step forward in this respect was taken by Scheirer who demonstrated that the perceived rhythmic content of many music types remains the same if only the power envelopes of a few subbands are preserved and then used to modulate a white noise signal [20]. Approximately five subbands were reported to suffice. Scheirer proposed a method where periodicity analysis was carried out within the subbands and the results were then combined across bands.

Although Scheirer's method was indeed very successful, a problem with it is that it applies primarily to music with a "strong beat." Harmonic changes for example in classical or vocal music go easily unnoticed using only a few subbands. In order to detect harmonic changes and note beginnings in *legato*[2] passages, approximately 40 logarithmically-distributed subbands would be needed.[3] However, this leads to a dilemma: the resolution is sufficient to distinguish harmonic changes but measuring periodicity at each narrow band separately is no longer appropriate. The power envelopes of individual narrow bands are not guaranteed to reveal the correct metrical periods—or even to show periodicity at all, because individual events may occupy different frequency bands.

To overcome the previous problem, consider another state-of-the-art system, that of Goto and Muraoka [18]. They detect narrow-band frequency components and sum their power differentials across predefined frequency ranges *before* onset detection and periodicity analysis takes place. This has the advantage that harmonic changes are detected, yet periodicity analysis takes place at wider bands.

There is a continuum between the previous two approaches. The tradeoff is: how many adjacent subbands are combined before the periodicity analysis and how many at the later stage when the bandwise periodicity analysis results are combined. In the following, we propose a method which can be seen as a synthesis of the approaches of Scheirer and Goto *et al.*.

Acoustic input signals are sampled at 44.1-kHz rate and 16-b resolution and then normalized to have zero mean and unity variance. Discrete Fourier transforms (DFTs) are calculated in successive 23-ms time frames which are Hanning-windowed and overlap 50%. In each frame, 36 triangular-response bandpass filters are simulated that are uniformly distributed on a critical-band scale between 50 Hz and 20 kHz [31, p. 176]. The power at each band is calculated and stored to $x_b(k)$, where k is the frame index and $b = 1, 2, \ldots, b_0$ is the band index, with $b_0 = 36$. The exact number of subbands is not critical.

There are many potential ways of measuring the degree of change in the power envelopes at critical bands. For humans, the smallest detectable change in intensity ΔI is approximately proportional to the intensity I of the signal, the same amount of increase being more prominent in a quiet signal. That is, $\Delta I/I$, the Weber fraction, is approximately constant perceptually [31, p. 134]. This relationship holds for intensities from about 20 dB to about 100 dB above the absolute hearing threshold. Thus, it is reasonable to normalize the differential of power with power, leading to $(d/dt)x_b(k)/x_b(k)$ which is equal to $(d/dt)\ln(x_b(k))$. This measures spectral change and can be seen to approximate the differential of *loudness*, since the perception of loudness for steady sounds is rougly proportional to the sum of log-powers at critical bands.

The logarithm and differentiation operations are both represented in a more flexible form. A numerically robust way of calculating the logarithm is the μ-law compression

$$y_b(k) = \frac{\ln(1 + \mu x_b(k))}{\ln(1 + \mu)} \tag{1}$$

which performs a logarithmic-like transformation for $x_b(k)$ as motivated above but behaves linearly near zero. The constant μ determines the degree of compression and can be used to adjust between a close-to-linear ($\mu < 0.1$) and a close-to-logarithmic ($\mu > 10^4$) transformation. The value $\mu = 100$ is employed, but all values in the range [10, 10^6] were found to perform almost equally well.

To achieve a better time resolution, the compressed power envelopes $y_b(k)$ are interpolated by factor two by adding zeros between the samples. This leads to the sampling rate $f_r = 172$ Hz. A sixth-order Butterworth low-pass filter with $f_{\mathrm{LP}} = 10$ Hz cutoff frequency is then applied to smooth the compressed and

[2]A smooth and connected style of playing in which no perceptible gaps are left between notes.

[3]In this case, the center frequencies are approximately one *whole tone* apart, which is the distance between, e.g., the notes c and d.

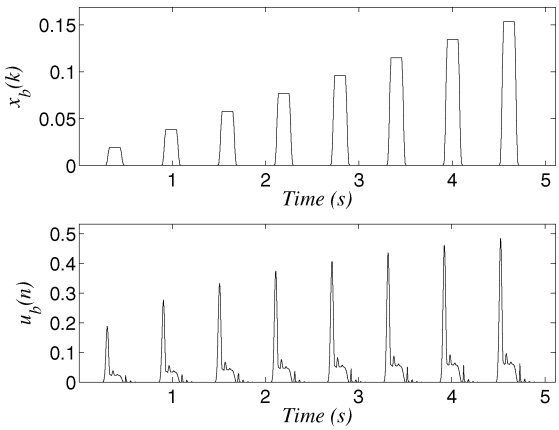


Fig. 3. Illustration of the dynamic compression and weighted differentiation steps for an artificial signal. Upper panel shows $x_b(k)$ and the lower panel shows $u_b(n)$.

interpolated power envelopes. The resulting smoothed signal is denoted by $z_b(n)$.

Differentiation of $z_b(n)$ is performed as follows. First, an HWR differential of $z_b(n)$ is calculated as

$$z_b'(n) = \text{HWR}(z_b(n) - z_b(n-1)) \tag{2}$$

where the function $\text{HWR}(x) = \max(x, 0)$ sets negative values to zero and is essential to make the differentiation useful. Then a weighted average of $z_b(n)$ and its differential $z_b'(n)$ is formed as

$$u_b(n) = (1-\lambda)z_b(n) + \lambda\frac{f_r}{f_{\text{LP}}}z_b'(n) \tag{3}$$

where $0 \leq \lambda \leq 1$ determines the balance between $z_b(n)$ and $z_b'(n)$, and the factor f_r/f_{LP} compensates for the fact that the differential of a low-pass-filtered signal is small in amplitude. A prototypical meter analysis system and a subset of our acoustic database (see Section III) were used to thoroughly investigate the effect of λ. Values between 0.6 and 1.0 performed well and $\lambda = 0.8$ was taken into use. Using this value instead of 1.0 makes a slight but consistent improvement in the analysis accuracy.

Fig. 3 illustrates the described dynamic compression and weighted differentiation steps for an artificial subband-power signal $x_b(k)$. Although the present work is motivated purely from a practical application point of view, it is interesting to note that the graphs in Fig. 3 bear considerable resemblance to the response of Meddis's auditory-nerve model to acoustic stimulation [32].

Finally, each m_0 adjacent bands are linearly summed to get $c_0 = \lceil b_0/m_0 \rceil$ accent signals at different frequency ranges c

$$v_c(n) = \sum_{b=(c-1)m_0+1}^{cm_0} u_b(n), \quad c = 1, \ldots, c_0. \tag{4}$$

The accent signals $v_c(n)$ serve as an intermediate data representation for musical meter analysis. They represent the degree of musical accent as a function of time at the wider frequency bands (channels) c. We use $b_0 = 36$ and $m_0 = 9$, leading to $c_0 = 4$.

It should be noted that combining each m_0 adjacent bands at this stage is not primarily an issue of computational complexity, but improves the analysis accuracy. Again, a prototypical meter analysis system was used to investigate the effect of different values of m_0. It turned out that neither of the extreme values $m_0 = b_0$ or $m_0 = 1$ is optimal, but using a large number of initial bands $b_0 > 20$ and three or four "accent bands" (channels) c_0 leads to the most reliable meter analysis. Other parameters were re-estimated in each case to ensure that this was not merely a symptom of parameter couplings. Elsewhere, at least Scheirer [20] and Laroche [22] have noted that a single accent signal (the case $m_0 = b_0$) appears not to be sufficient as an intermediate representation for rhythmic parsing.

The presented form of calculating the bandwise accent signals is very flexible when varying μ, λ, b_0, and m_0. A representation similar to that used by Scheirer in [20] is obtained by setting $\mu = 0.1$, $\lambda = 1$, $b_0 = 6$, $m_0 = 1$. A representation roughly similar to that used by Goto in [18] is obtained by setting $\mu = 0.1$, $\lambda = 1$, $b_0 = 36$, $m_0 = 6$. In the following, the fixed values $\mu = 100$, $\lambda = 0.8$, $b_0 = 36$, $m_0 = 9$ are used.

B. Bank of Comb Filter Resonators

Periodicity of the bandwise accent signals $v_c(n)$ is analyzed to estimate the *salience* (weight) of different pulse period candidates. Four different period estimation algorithms were evaluated: a method based on autocorrelation, another based on the method of de Cheveigné and Kawahara [33], different types of comb-filter resonators [20], and banks of phase-locking resonators [14].

As an important observation, three of the four period estimation methods performed equally well after a thorough optimization. This suggests that the key problems in meter analysis are in measuring the degree of musical accentuation and in modeling higher level musical knowledge, not in finding exactly the correct period estimator. The period estimation method presented in the following was selected because it is by far the least complex among the three best-performing algorithms, requiring only few parameters and no additional postprocessing steps.

Using a bank of comb-filter resonators with a constant half-time was originally proposed for tactus tracking by Scheirer [20]. The comb filters that we use have an exponentially-decaying impulse response where the *half-time* refers to the delay during which the response decays to a half of its initial value. The output of a comb filter with delay τ for input $v_c(n)$ is given by

$$r_c(\tau, n) = \alpha_\tau r_c(\tau, n-\tau) + (1-\alpha_\tau)v_c(n) \tag{5}$$

where the feedback gain $\alpha_\tau = 0.5^{\tau/T_0}$ is calculated based on a selected half-time T_0 in samples. We used a half-time equivalent to 3 s, i.e., $T_0 = 3.0 \text{ s} \cdot f_r$, which is short enough to react to tempo changes but long enough to reliably estimate pulse-periods of up to 4 s in length.

The comb filters implement a frequency response where the frequencies $kf_r/\tau, k = 0, \ldots, \lfloor \tau/2 \rfloor$ have a unity response and the maximum attenuation between the peaks is $((1-\alpha_\tau)/(1+$

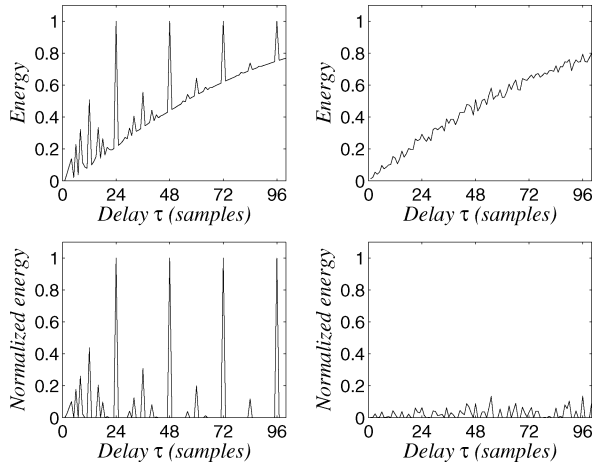


Fig. 4. Resonator energies for an impulse train with a period-length of 24 samples (left) and for white noise (right). Upper panels show the energies $\hat{r}_c(\tau, n)$ and the lower panels normalized energies $s_c(\tau, n)$.

$\alpha_\tau))^2$. The overall power $\gamma(\alpha_\tau)$ of a comb filter with feedback gain α_τ can be calculated by integrating over the squared impulse response, which yields

$$\gamma(\alpha_\tau) = \frac{(1-\alpha_\tau)^2}{1-\alpha_\tau^2}. \tag{6}$$

A bank of such resonators was applied, with τ getting values from 1 to $\tau_{\max}$, where $\tau_{\max} = 688$ corresponds to 4 s. The computational complexity of one resonator is $O(1)$ per input sample, and the overall resonator filterbank requires of the order $c_0 f_r \tau_{\max}$ operations per second, which is not too demanding for real-time applications.

Instantaneous energies $\hat{r}_c(\tau, n)$ of each comb filter in channel c at time n are calculated as

$$\hat{r}_c(\tau, n) = \frac{1}{\tau} \sum_{i=n-\tau+1}^{n} r_c(\tau, i)^2. \tag{7}$$

These are then normalized to obtain

$$s_c(\tau, n) = \frac{1}{1-\gamma(\alpha_\tau)} \left(\frac{\hat{r}_c(\tau, n)}{\hat{v}_c(n)} - \gamma(\alpha_\tau) \right) \tag{8}$$

where $\hat{v}_c(n)$ is the energy of the accent signal $v_c(n)$, calculated by squaring $v_c(n)$ and by applying a leaky integrator, i.e., a resonator which has $\tau = 1$ and the same three-second half-time as the other resonators. Normalization with $\gamma(\alpha_\tau)$ compensates for the differences in the overall power responses for different α_τ. The proposed normalization is advantageous because it preserves a unity response at the peak frequencies and at the same time removes a τ-dependent trend for a white-noise input.

Fig. 4 shows the resonator energies $\hat{r}_c(\tau, n)/\hat{v}_c(n)$ and the normalized energies $s_c(\tau, n)$ for two types of artificial input $v_c(n)$: an impulse train and a white-noise signal. It is important to notice that all resonators that are in rational-number relations to the period of the impulse train (24 samples) show response to it. In the case of the autocorrelation function, for example, only integer multiples of 24 come up and an explicit postprocessing step was necessary to generate responses to the subharmonic lags and to achieve the same meter analysis performance. This step is not needed for comb filter resonators where the conceptual complexity and the number of free parameters, thus, remains smaller.

Finally, a function $s(\tau, n)$ which represents the overall saliences of different metrical pulses at time n is obtained as

$$s(\tau, n) = \sum_{c=1}^{c_0} s_c(\tau, n). \tag{9}$$

This function acts as the *observation* for the probabilistic model that estimates the pulse periods.

For tatum period estimation, the discrete power spectrum $S(f, n)$ of $s(\tau, n)$ is calculated as

$$S(f, n) = f \left| \frac{1}{\tau_{\max}} \sum_{\tau=1}^{\tau_{\max}} \left(s(\tau, n) \zeta(\tau) e^{-i2\pi f(\tau-1)/\tau_{\max}} \right) \right|^2 \tag{10}$$

where the emphasis with f compensates for a spectral trend and the window function $\zeta(\tau)$ is half-Hanning

$$\zeta(\tau) = 0.5 \left(1 - \cos \left(\frac{\pi(\tau - 1 + \tau_{\max})}{\tau_{\max}} \right) \right). \tag{11}$$

The rationale behind calculating the DFT in (10) is that, by definition, other pulse periods are integer multiples of the tatum period. Thus, the overall function $s(\tau, n)$ contains information about the tatum and this is conveniently gathered for each tatum-frequency candidate f using the DFT as in (10). For comparison, Gouyon *et al.* [24] used an inter-onset-interval histogram and Maher's two-way mismatch procedure [34] served the same purpose. Their idea was to find a tatum period which best explained the multiple harmonically related peaks in the histogram. Frequencies above 20 Hz can be discarded from $S(f, n)$, since tatum frequencies faster than this are very rare.

It should be noted that the observation $s(\tau, n)$ and its spectrum $S(f, n)$ are zero-phase, meaning that the phases of the pulses at different metrical levels have to be estimated using some other source of information. As will be discussed in Section II-D, the phases are estimated based on the states of the comb filters, after the periods have been decided first.

C. Probabilistic Model for Pulse Periods

Period-lengths of the metrical pulses can be estimated independently of their phases and it is reasonable to compute the phase only for the few winning periods.[4] Thus, the proposed method finds periods first and then the phases (see Fig. 2). Although estimating the phases is not trivial, the search problem is largely completed when the period-lengths have been found.

Musical meter cannot be assumed to remain static over the whole duration of a piece. It has to be estimated causally at successive time instants and there must be some tying between the successive estimates. Also, the dependencies between different metrical pulse levels have to be taken into account. These require prior musical knowledge which is encoded in the probabilistic model to be presented.

[4]For comparison, Laroche [22] estimates periods and phases simultaneously, at the expense of a larger search space. Here three pulse levels are being estimated jointly and estimating periods and phases separately serves the purpose of retaining a moderately-sized search space.

For period estimation, a hidden Markov model that describes the simultaneous evolution of four processes is constructed. The observable variable is the vector of instantaneous energies of the resonators, $s(\tau, n)$, denoted $\boldsymbol{s}_n$ in the following. The unobservable processes and the corresponding hidden variables are the tatum period τ_n^{A}, tactus period τ_n^{B}, and measure period τ_n^{C}. As a mnemonic for this notation, recall that the tatum is the temporally atomic (A) pulse level, the tactus pulse is often called "beat" (B), and the musical measure pulse is related to the harmonic (i.e., chord) change rate (C). For convenience, we use $\boldsymbol{q}_n = [j, k, l]$ to denote a "meter state," equivalent to $\tau_n^{\mathrm{A}} = j$, $\tau_n^{\mathrm{B}} = k$, and $\tau_n^{\mathrm{C}} = \mathrm{l}$. The hidden state process is a time-homogenous first-order Markov model which has an initial state distribution $P(\boldsymbol{q}_1)$ and transition probabilities $P(\boldsymbol{q}_n \mid \boldsymbol{q}_{n-1})$. The observable variable is conditioned only on the current state, i.e., we have the state-conditional observation densities $p(\boldsymbol{s}_n \mid \boldsymbol{q}_n)$.

The joint probability density of a state sequence $Q = (\boldsymbol{q}_1 \boldsymbol{q}_2 \ldots \boldsymbol{q}_N)$ and observation sequence $O = (\boldsymbol{s}_1 \boldsymbol{s}_1 \ldots \boldsymbol{s}_N)$ can be written as

$$p(Q, O) = P(\boldsymbol{q}_1) p(\boldsymbol{s}_1 \mid \boldsymbol{q}_1) \prod_{n=2}^{N} P(\boldsymbol{q}_n \mid \boldsymbol{q}_{n-1}) p(\boldsymbol{s}_n \mid \boldsymbol{q}_n) \tag{12}$$

where the term $P(\boldsymbol{q}_n \mid \boldsymbol{q}_{n-1})$ can be decomposed as

$$\begin{aligned} &P(\boldsymbol{q}_n \mid \boldsymbol{q}_{n-1}) \\ &= P(\tau_n^{\mathrm{B}} \mid \boldsymbol{q}_{n-1}) P(\tau_n^{\mathrm{A}} \mid \tau_n^{\mathrm{B}}, \boldsymbol{q}_{n-1}) P(\tau_n^{\mathrm{C}} \mid \tau_n^{\mathrm{B}}, \tau_n^{\mathrm{A}}, \boldsymbol{q}_{n-1}). \end{aligned} \tag{13}$$

It is musically meaningful to assume that

$$P(\tau_n^{\mathrm{C}} \mid \tau_n^{\mathrm{B}}, \tau_n^{\mathrm{A}}, \boldsymbol{q}_{n-1}) = P(\tau_n^{\mathrm{C}} \mid \tau_n^{\mathrm{B}}, \boldsymbol{q}_{n-1}) \tag{14}$$

i.e., given the tactus period, the tatum period does not give additional information regarding the measure period. We further assume that given τ_{n-1}^{B}, the other two hidden variables at time $n-1$ give no additional information regarding τ_n^{B}. For the tatum and measure periods $\tau_n^i, i \in \{\mathrm{A}, \mathrm{C}\}$, we assume that given τ_{n-1}^i and τ_n^{B}, the other two hidden variables at time $n-1$ give no additional information regarding τ_n^i. It follows that (13) can be written as

$$\begin{aligned} &P(\boldsymbol{q}_n \mid \boldsymbol{q}_{n-1}) \\ &\quad = P(\tau_n^{\mathrm{B}} \mid \tau_{n-1}^{\mathrm{B}}) P(\tau_n^{\mathrm{A}} \mid \tau_n^{\mathrm{B}}, \tau_{n-1}^{\mathrm{A}}) P(\tau_n^{\mathrm{C}} \mid \tau_n^{\mathrm{B}}, \tau_{n-1}^{\mathrm{C}}). \end{aligned} \tag{15}$$

Using the same assumptions, $P(\boldsymbol{q}_1)$ is decomposed and simplified as

$$P(\boldsymbol{q}_1) = P(\tau_1^{\mathrm{B}}) P(\tau_1^{\mathrm{A}} \mid \tau_1^{\mathrm{B}}) P(\tau_1^{\mathrm{C}} \mid \tau_1^{\mathrm{B}}). \tag{16}$$

The described modeling assumptions lead to a structure which is represented as a directed acyclic graph in Fig. 5. The arrows in the graph represent conditional dependencies between the variables. The circles denote hidden variables and the observed variable is marked with boxes. The tactus pulse has a central role in meter perception and it is not by chance that the other two variables are drawn to depend on it [1, pp. 73–74]. The assumption in (14) is not valid if the variables are permuted.

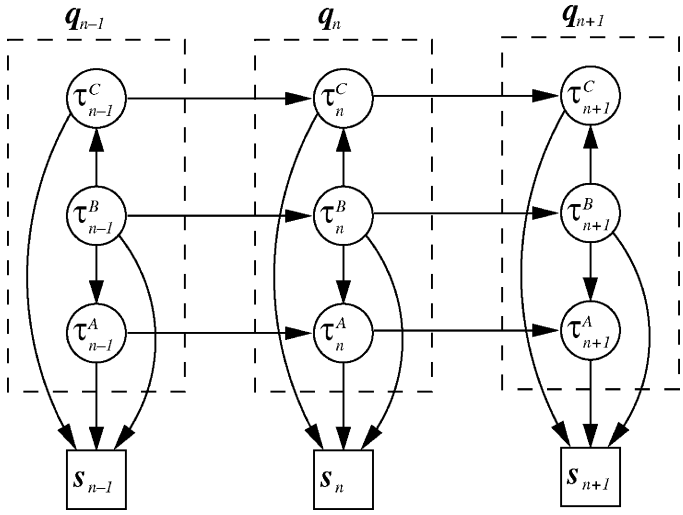


Fig. 5. Hidden Markov model for the temporal evolution of the tatum, beat, and measure pulse periods.

1) Estimation of the State-Conditional Observation Likelihoods: The remaining problem is to find reasonable estimates for the model parameters, i.e., for the probabilities that appear in (12)–(16). In the following, we ignore the time indexes for a while for simplicity. The state-conditional observation likelihoods $p(\boldsymbol{s} \mid \boldsymbol{q})$ are estimated from a database of musical recordings where the musical meter has been hand-labeled (see Section III). However, the data is very limited in size compared to the number of parameters to be estimated. Estimation of the state densities for each different $\boldsymbol{q} = [j, k, l]$ is impossible since each of the three discrete hidden variables can take on several hundreds of different values. By making a series of assumptions we arrive at the following approximation for $p(\boldsymbol{s} \mid \boldsymbol{q})$:

$$p(\boldsymbol{s} \mid \boldsymbol{q} = [j, k, l]) \propto s(k) s(l) S\left(\frac{1}{j}\right) \tag{17}$$

where $s(\tau)$ and $S(f)$ are as defined in (9)–(10), omitting the time indexes. The Appendix presents the derivation of (17) and the underlying assumptions in detail. An intuitive rationale of (17) is that a truly existing tactus or measure pulse appears as a peak in $s(\tau)$ at the lag that corresponds to the pulse period. Analogously, the tatum period appears as a peak in $S(f)$ at the frequency that corresponds to the inverse of the period. The product of these three values correlates approximately linearly with the likelihood of the observation given the meter.

2) Estimation of the Transition and Initial Probabilities: In (15), the term $P(\tau_n^{\mathrm{A}} \mid \tau_n^{\mathrm{B}}, \tau_{n-1}^{\mathrm{A}})$ can be decomposed as

$$P(\tau_n^{\mathrm{A}} \mid \tau_n^{\mathrm{B}}, \tau_{n-1}^{\mathrm{A}}) = P(\tau_n^{\mathrm{A}} \mid \tau_{n-1}^{\mathrm{A}}) \frac{P(\tau_n^{\mathrm{A}}, \tau_n^{\mathrm{B}} \mid \tau_{n-1}^{\mathrm{A}})}{P(\tau_n^{\mathrm{A}} \mid \tau_{n-1}^{\mathrm{A}}) P(\tau_n^{\mathrm{B}} \mid \tau_{n-1}^{\mathrm{A}})} \tag{18}$$

where the first factor represents transition probabilities between successive period estimates and the second term represents the

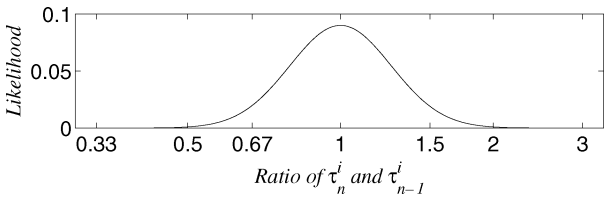


Fig. 6. Likelihood function $f(\tau_n^i/\tau_{n-1}^i)$ which describes the tendency that the periods are slowly-varying.

relation dependencies of simultaneous periods, τ_n^{A} and τ_n^{B}, independent of their actual frequencies of occurrence (in practice τ_n^{B} tends to be integer multiple of τ_n^{A}). Similarly

$$P(\tau_n^{\mathrm{C}}\,|\,\tau_n^{\mathrm{B}},\tau_{n-1}^{\mathrm{C}}) = P(\tau_n^{\mathrm{C}}\,|\,\tau_{n-1}^{\mathrm{C}})\frac{P(\tau_n^{\mathrm{C}},\tau_n^{\mathrm{B}}\,|\,\tau_{n-1}^{\mathrm{C}})}{P(\tau_n^{\mathrm{C}}\,|\,\tau_{n-1}^{\mathrm{C}})P(\tau_n^{\mathrm{B}}\,|\,\tau_{n-1}^{\mathrm{C}})}. \tag{19}$$

The transition probabilities $P(\tau_n^i \mid \tau_{n-1}^i)$, $i \in \{\mathrm{A},\mathrm{B},\mathrm{C}\}$ between successive period estimates are obtained as follows. Again, the number of possible transitions is too large for any reasonable estimates to be obtained by counting occurrences. The transition probability is modeled as a product of the prior probability for a certain period, $P(\tau_1^i)$, and a term $f(\tau_n^i/\tau_{n-1}^i)$ which describes the tendency that the periods are slowly-varying

$$P(\tau_n^i \mid \tau_{n-1}^i) = P(\tau_1^i)\frac{P(\tau_n^i,\tau_{n-1}^i)}{P(\tau_n^i)P(\tau_{n-1}^i)} \approx P(\tau_1^i) f\left(\frac{\tau_n^i}{\tau_{n-1}^i}\right) \tag{20}$$

where $i \in \{\mathrm{A},\mathrm{B},\mathrm{C}\}$. The function f

$$f\left(\frac{\tau_n^i}{\tau_{n-1}^i}\right) = \frac{1}{\sigma_1\sqrt{2\pi}}\exp\left[-\frac{1}{2\sigma_1^2}\left(\ln\left(\frac{\tau_n^i}{\tau_{n-1}^i}\right)\right)^2\right] \tag{21}$$

implements a normal distribution as a function of the logarithm of the ratio of successive period values. It follows that the likelihood of large changes in period is higher for long periods, and that period doubling and halving are equally probable. The parameter $\sigma_1 = 0.2$ was found by monitoring the performance of the system in simulations. The distribution (21) is illustrated in Fig. 6.[5]

Prior probabilities for tactus period lengths, $P(\tau^{\mathrm{B}})$, have been measured from actual data by several authors [12], [35], [36]. As suggested by Parncutt [12], we apply the two-parameter lognormal distribution

$$p(\tau^i) = \frac{1}{\tau^i\sigma^i\sqrt{2\pi}}\exp\left[-\frac{1}{2(\sigma^i)^2}\left(\ln\left(\frac{\tau^i}{m^i}\right)\right)^2\right] \tag{22}$$

where m^i and σ^i are the scale and shape parameters, respectively. For the tactus period, the values $m^{\mathrm{B}} = 0.55$ and $\sigma^{\mathrm{B}} = 0.28$ were estimated by counting the occurrences of different period lengths in our hand-labeled database (see Section III) and by fitting the lognormal distribution to the histogram data. The parameters depend somewhat on genre [35], [36] but since the genre is generally not known, common parameter values are used here. Fig. 7 shows the period-length histograms and the corresponding lognormal distributions for the tactus, measure, and tatum periods. The scale and shape parameters for the tatum and measure periods are $m^{\mathrm{A}} = 0.18$, $\sigma^{\mathrm{A}} = 0.39$, $m^{\mathrm{C}} = 2.1$, and $\sigma^{\mathrm{C}} = 0.26$, respectively. These were estimated from the hand-labeled data in the same way.

[5]For comparison, Laroche uses a cost function where tempo changes exceeding a certain threshold are assigned a fixed cost and smaller tempo changes cause no cost at all [22].

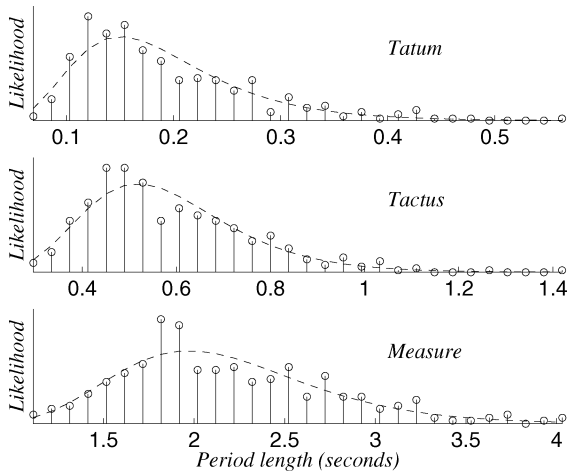


Fig. 7. Period-length histograms and the corresponding lognormal distributions for tatum, tactus, and measure pulses.

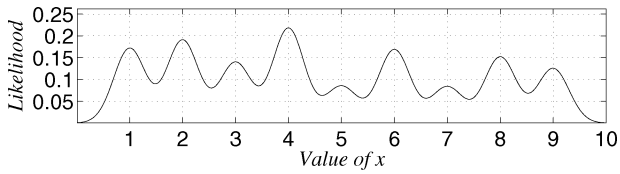


Fig. 8. Distribution $g(x)$ which models the relation dependencies of simultaneous periods [see (25)].

The relation dependencies of simultaneous periods are modeled as follows. We model the latter terms in (18)–(19) as

$$\frac{P(\tau_n^{\mathrm{A}},\tau_n^{\mathrm{B}} \mid \tau_{n-1}^{\mathrm{A}})}{P(\tau_n^{\mathrm{A}} \mid \tau_{n-1}^{\mathrm{A}})P(\tau_n^{\mathrm{B}} \mid \tau_{n-1}^{\mathrm{A}})} \approx g\left(\frac{\tau^{\mathrm{B}}}{\tau^{\mathrm{A}}}\right) \tag{23}$$

$$\frac{P(\tau_n^{\mathrm{C}},\tau_n^{\mathrm{B}} \mid \tau_{n-1}^{\mathrm{C}})}{P(\tau_n^{\mathrm{C}} \mid \tau_{n-1}^{\mathrm{C}})P(\tau_n^{\mathrm{B}} \mid \tau_{n-1}^{\mathrm{C}})} \approx g\left(\frac{\tau^{\mathrm{C}}}{\tau^{\mathrm{B}}}\right) \tag{24}$$

where $g(x)$ is a Gaussian mixture density of the form

$$g(x) = \sum_{l=1}^{9} w_l N(x; l, \sigma_2) \tag{25}$$

where w_l are the component weights and sum to unity, l are the component means, and $\sigma_2 = 0.3$ is the common variance. The function models the relation dependencies of simultaneous periods, independent of their actual frequencies of occurrence. The exact weight values are not critical, but are designed to realize a tendency toward binary or ternary integer relationships between concurrent pulses. For example, it happens quite often that one tactus period consists of two, four, or six tatum periods, but multiples five and seven are much less likely in music and, thus, have lower weights. The distribution is shown in Fig. 8. The Gaussian mixture model was employed to allow some deviation from strictly integral ratios. In theory, the period-lengths should be precisely in integral ratios but, in practice, there are inaccuracies since the period candidates are chosen from discrete vectors $\boldsymbol{s}_n$ and $\boldsymbol{S}_n$. These inaccuracies are conveniently handled

by choosing an appropriate value for σ_2 in the previous model. The weights w_l were obtained by first assigning them values according to a musical intuition. Then the dynamic range of the weights was found by raising them to a common power which was varied between 0.1 and 10. The value which performed best in small-scale simulations was selected. Finally, small adjustments to the values were made.

It should be noted that here the model parameters were specified in part by hand, considering one probability distribution at a time. It seems possible to devise an algorithm that would learn the model parameters jointly by Bayesian optimization, that is, by maximizing the posterior probability of training data given the prior distributions. However, even after all the described modeling assumptions and simplifications, deriving an expectation-maximization algorithm [37] for the described model, for example, is not easy and such an algorithm does not exist at the present time.

3) Finding the Optimal Sequence of Period Estimates: Now we must obtain an estimate for the unobserved state variables given the observed resonator energies and the model parameters. We do this by finding the most likely sequence of state variables $Q = (\boldsymbol{q}_1 \boldsymbol{q}_2 \ldots \boldsymbol{q}_N)$ given the observed data $O = (\boldsymbol{s}_1 \boldsymbol{s}_1 \ldots \boldsymbol{s}_N)$. This can be straighforwardly computed using the Viterbi algorithm widely applied in speech recognition [38]. Thus, we seek the sequence of period estimates

$$\hat{Q} = \arg\max_{Q} (p(Q, O)) \tag{26}$$

where $p(Q, O)$ denotes the joint probability density of the hidden and observed variables (see (12)).

In a causal model, the meter estimate $\boldsymbol{q}_n$ at time n is determined according to the end-state of the best partial path at that point in time. A noncausal estimate after seeing a complete sequence of observations can be computed using backward decoding.

Evaluating all the possible path candidates would be computationally very demanding. Therefore, we apply a suboptimal beam-search strategy and evaluate only a predefined number of the most promising path candidates at each time instant. The selection of the most promising candidates is made using a greedy selection strategy. Once in a second, we select K best candidates independently for the tatum, tactus, and measure periods. The number of candidates $K = 5$ was found to be safe and was used in simulations. The selection is made by maximizing $p(\tau_n^i) p(\boldsymbol{s}_n \mid \tau_n^i)$ for $i \in \{\mathrm{A}, \mathrm{B}, \mathrm{C}\}$. The probabilities in (23)–(24) could be included to ensure that the selected candidates are consistent with each other, but in practice this is unnecessary. After selecting the best candidates for each, we need only to compute the observation likelihoods for $K^3 = 125$ meter candidates, i.e., for the different combinations of the tatum, tactus, and measure periods. This is done according to (17) and the results are stored into a data vector. The transition probabilities are computed using (15) and stored into a 125-by-125 matrix. These data structures are then used in the Viterbi algorithm.

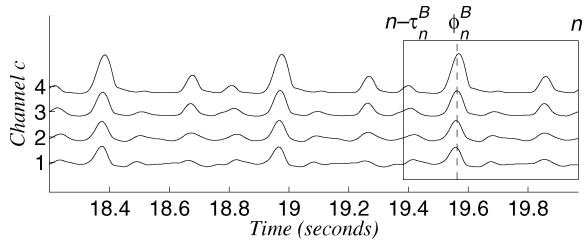


Fig. 9. Rectangle indicates the observation matrix R_n^{B} for tactus phase estimation at time n (here period τ_n^{B} is 0.51 s.). Dashed line shows the correct phase in this case.

D. Phase Estimation

The phases of the three pulses are estimated at successive time instants, after the periods have been decided at these points. We use $\hat{\tau}_n^i$, $i \in \{\mathrm{A}, \mathrm{B}, \mathrm{C}\}$ to refer to the estimated periods of the tatum, tactus, and measure pulses at time n, respectively. The corresponding phases of the three pulses, φ_n^i, are expressed as "temporal anchors," i.e., time values when the nearest beat unit occurs with respect to the beginning of a piece. The periods and phases τ_n^i and φ_n^i completely define the meter at time n.

In principle, the phase of the measure pulse, φ_n^{C}, determines the phases of all the three levels. This is because in a well-formed meter each measure-level beat must coincide with a beat at all the lower metrical levels. However, determining the phase of the measure pulse is difficult and turned out to require rhythmic pattern matching techniques, whereas tactus phase estimation is more straightforward and robust. We therefore propose a model where the tactus and measure phases are estimated separately using two parallel models. For the tatum pulse, phase estimation is not needed but the tactus phase can be used.

Scheirer proposed using the state vectors of comb filters to determine the phase of the tactus pulse [20]. This is equivalent to using the latest τ outputs of a resonator with delay τ. We have resonators at several channels c and, consequently, an output matrix $r_c(\tau, j)$ where $c = 1, 2, \ldots, c_0$ is the channel index and the phase index j takes on values between $n - \tau + 1$ and n when estimation is taking place at time n. For convenience, we use R_n^i to denote the output matrix $r_c(\hat{\tau}_n^i, j)$ of a found pulse period $\hat{\tau}_n^i$ and the notation $(R_n^i)_{c,j}$ to refer to the individual elements of R_n^i. The matrix R_n^i acts as the observation for phase estimation at time n.

Fig. 9 shows an example of the observation matrix R_n^{B} when tactus phase estimation is taking place 20 s after the beginning of a piece. The four signals at different channels are the outputs of the comb filter which corresponds to the estimated tactus period $\hat{\tau}_n^{\mathrm{B}} = 0.51$ s. The output matrix R_n^{B} contains the latest 0.51 s of the output signals, as indicated with the rectangle. The correct phase φ_n^{B} is marked with a dashed line.

Two separate hidden Markov models are evaluated in parallel, one for the tactus phase and another for the measure phase. No joint estimation is attempted. The two models are very similar and differ only in how the state-conditional observation densities are defined. In both models, the observable variable is the output matrix R_n^i of the resonator $\hat{\tau}_n^i$ which corresponds to the found pulse period. The hidden variable is the phase of the pulse, φ_n^i, taking on values between $n - \hat{\tau}_n^i + 1$ and n. The hidden state process is a time-homogenous first-order Markov model which

has an initial state distribution $P(\varphi_1)$ and transition probabilities $P(\varphi_n \mid \varphi_{n-1})$. The observable variable is conditional only on the current state, thus, we have the state-conditional observation densities $p(R_n^i \mid \varphi_n^i)$.

Again, the remaining problem is to find reasonable estimates for the model parameters. State-conditional observation likelihoods $p(R_n^{\mathrm{B}} \mid \varphi_n^{\mathrm{B}})$ for the tactus pulse are approximated as

$$p(R_n^{\mathrm{B}} \mid \varphi_n^{\mathrm{B}} = j) \propto \sum_{c=1}^{c_0} (c_0 - c + 2)(R_n^{\mathrm{B}})_{c,j} \tag{27}$$

where $c = 1$ corresponds to the lowest frequency channel. That is, the likelihood is proportional to a weighted sum of the resonator outputs across the channels. Across-band summing is intuitively meaningful and earlier used in [20] and [30]. Emphasizing the low frequencies is motivated by the "stable bass" rule as stated in [1], and improved the robustness of phase estimation in simulations. The exact weight values are not critical.

For the purpose of estimating the phase of the measure pulse, a formula for the state-conditional observation likelihoods analogous to that in (27) is derived, but so that different channels are weighted and delayed in a more complex manner. It turned out that rhythmic pattern matching of some kind is necessary to analyze music at this time scale and to estimate the measure phase φ_n^{C} based on the output matrix R_n^{C}. That is, no simple formula such as (27) exists. The drawback of this is that rhythmic pattern matching is more genre-specific than for example the stable bass rule which appears to be quite universal. In the case that the system would have access to the pitch content of an incoming piece, the points of harmonic change might serve as cues for estimating the measure phase in a more straightforward manner. However, this remains to be proved. Estimation of the higher level metrical pulses in audio data has been earlier attempted by Goto and Muraoka who resorted to pattern matching [18] or to straightforward chord change detection [19]. The method presented in the following is the most reliable that we found.

First, a vector $h_n(l)$ is constructed as

$$h_n(l) = \sum_{c=1}^{c_0} \sum_{k=0}^{3} \eta_{c,k} (R_n^{\mathrm{C}})_{c,j(k,l,n)} \tag{28}$$

where

$$l = 0, 1, \ldots, \hat{\tau}_n^{\mathrm{C}} - 1 \tag{29}$$

$$j(k,l,n) = n - \hat{\tau}_n^{\mathrm{C}} + 1 + \left(\left(l + \frac{k\hat{\tau}_n^{\mathrm{C}}}{4} \right) \bmod \hat{\tau}_n^{\mathrm{C}} \right) \tag{30}$$

and $(x \bmod y)$ denotes modulus after division. The scalars $\eta_{c,k}$ are weights for the resonator outputs at channels c and with delays k. The weights $\eta_{c,k}$ are used to encode a typical pattern of energy fluctuations within one measure period, so that the maximum of $h_n(l)$ indicates the measure phase. The delay k is expressed in quarter-measure units so that k corresponds to the delay $k\hat{\tau}_n^{\mathrm{C}}/4$. For example, a simple pattern consisting of two events, a low-frequency event (at channel $c = 1$) in the beginning of a measure ($k = 0$) and a loud event in the middle of the measure ($k = 2$), could be represented by defining the weights $\eta_{1,0} = 3$ (low), $\eta_{c,2} = 1$ for all c (loud), and $\eta_{c,k} = 0$ otherwise.

Two rhythmic patterns were found that generalized quite well over our database. The weight matrices $\eta_{c,k}^{(1)}$ and $\eta_{c,k}^{(2)}$ of these patterns are given in the Appendix and lead to the corresponding $h_n^{(1)}(l)$ and $h_n^{(2)}(l)$. The patterns were found by trial and error, trying out various arrangements of simple atomic events and monitoring the behavior of $h_n(l)$ against manually annotated phase values. Both of the two patterns can be characterized as a pendulous motion between a low-frequency event and a high-intensity event. The first pattern can be summarized as "low, loud, –, loud," and the second as "low, –, loud, –." The two patterns are combined into a single vector to perform phase estimation according to whichever pattern matches better to the data

$$h_n^{(1,2)}(l) = \max\left(h_n^{(1)}(l), h_n^{(2)}(l) \right). \tag{31}$$

The state-conditional observation likelihoods are then defined as

$$p(R_n^{\mathrm{C}} \mid \varphi_n^{\mathrm{C}} = j) \propto h_n^{(1,2)}(j - (n - \hat{\tau}_n^{\mathrm{C}} + 1)). \tag{32}$$

Obviously, the two patterns imply a *binary time signature*: they assume that one measure period consists of two or four tactus periods. Analysis results for ternary meters will be separately discussed in Section III-C.

Other pattern-matching approaches were evaluated, too. In particular, we attempted to sample R_n^{C} at the times of the tactus beats and to train a statistical classifier to choose the beat which corresponds to the measure beat (see [36] for further elaboration on this idea). However, the methods were basically equivalent to that described previously, yet less straightforward to implement and performed slightly worse.

Transition probabilities $P(\varphi_n^i \mid \varphi_{n-1}^i)$ between successive phase estimates are modeled as follows. Given two phase estimates (i.e., beat occurrence times), the conditional probability which ties the successive estimates is assumed to be normally distributed as a function of a *prediction error* e which measures the deviation of φ_n^i from the predicted next beat occurence time given the previous beat time φ_{n-1}^i and the period $\hat{\tau}_n^i$

$$P(\varphi_n^i \mid \varphi_{n-1}^i) = \frac{1}{\sigma_3\sqrt{2\pi}} \exp\left(-\frac{e^2}{2\sigma_3^2} \right) \tag{33}$$

where

$$e = \frac{1}{\hat{\tau}_n^i} \left\{ \left[\left(\left| \varphi_n^i - \varphi_{n-1}^i \right| + \frac{\hat{\tau}_n^i}{2} \right) \bmod \hat{\tau}_n^i \right] - \frac{\hat{\tau}_n^i}{2} \right\} \tag{34}$$

and $\sigma_3 = 0.1$ is common for $i \in \{\mathrm{B}, \mathrm{C}\}$. In (34), it should be noted that any integer number of periods $\hat{\tau}_n^i$ may elapse between φ_{n-1}^i and φ_n^i. Since estimates are produced quite frequently compared to the pulse rates, in many cases $\varphi_n^i = \varphi_{n-1}^i$. The initial state distributions $P(\varphi_1^i)$ are assumed to be uniform.

Using (27), (32), and (33), causal and noncausal computation of phase is performed using the Viterbi algorithm as described in Section II-C. Fifteen phase candidates for both the winning tactus and the winning measure period are generated once in a second. The candidates are selected in a greedy manner by picking local maxima in $p(R_n^i \mid \varphi_n^i = j)$. The corresponding probability values are stored into a vector and transition probabilities between successive estimates are computed using (33).

E. Sound Onset Detection and Extrametrical Events

Detecting the beginnings of discrete acoustic events one-by-one has many uses. It is often of interest whether an event occurs at a metrical beat or not, and what is the exact timing of an event with respect to its ideal metrical position. Also, in some musical pieces there are extrametrical events, such as *triplets*, where an entity of, e.g., four tatum periods is exceptionally divided into three parts, or *grace notes* which are pitched events that occur shortly before a metrically stable event.

In this paper, we used an onset detector as a front-end to one of the reference systems (designed for MIDI input) to enable it to process acoustic signals. Rather robust onset detection is achieved by using an *overall accent signal* $v(n)$ which is computed by setting $m_0 = b_0$ in (4). Local maxima in $v(n)$ represent onset candidates and the value of $v(n)$ at these points reflects the likelihood that a discrete event occurred. A simple peak-picking algorithm with a fixed threshold level can then be used to distinguish genuine onsets from the changes and modulations that take place during the ringing of a sound. Automatic adaptation of the threshold would presumably further improve the detection accuracy.

III. Results

This section looks at the performance of the proposed method in simulations and compares the results with two reference systems. Also, the importance of different processing elements will be validated.

A. Experimental Setup

Table I shows the statistics of the database[6] that was used to evaluate the accuracy of the proposed meter analysis method and the two reference methods. Musical pieces were collected from CD recordings, downsampled to a single channel, and stored to a hard disc using 44.1-kHz sampling rate and 16-b resolution. The database was created for the purpose of musical signal classification in general and the balance between genres is according to an informal estimate of what people listen to.

The metrical pulses were manually annotated for approximately one-minute long excerpts which were selected to represent each piece. Tactus and measure-pulse annotations were made by a musician who tapped along with the pieces. The tapping signal was recorded and the tapped beat times were then detected semiautomatically using signal level thresholding. The tactus pulse could be annotated for 474 of a total of 505 pieces. The measure pulse could be reliably marked by listening for 320 pieces. In particular, annotation of the measure pulse was not attempted for classical music without the musical scores. Tatum pulse was annotated by the first author by listening to the pieces together with the annotated tactus pulse and by determining the integer ratio between the tactus and the tatum period lengths. The integer ratio was then used to interpolate the tatum beats between the tapped tactus beats.

Evaluating a meter analysis system is not trivial. The issue has been addressed in depth by Goto and Muraoka in [39]. As suggested by them, we use the longest *continuous* correctly analyzed segment as a basis for measuring the performance. This means that one inaccuracy in the middle of a piece leads to 50% performance. The longest continuous sequence of correct pulse estimates in each piece is sought and compared to the length of the segment which was given to be analyzed. The ratio of these two lengths determines the performance rate for one piece and these are then averaged over all pieces. However, prior to the meter analysis, all the algorithms under consideration were given a 4-s "build-up period" in order to make it theoretically possible to estimate the correct period already from the beginning of the evaluation segment. Also, it was taken care that none of the input material involved tempo discontinuities. More specifically, the interval between two tapped reference beat times (pulse period) does not change more than 40% at a time, between two successive beats. Other tempo fluctuations were naturally allowed.

TABLE I
STATISTICS OF THE EVALUATION DATABASE

	# Pieces with annotated pulses		
Genre	Tatum	Tactus	Measure
Classical	69	84	0
Electronic / dance	47	66	62
Hip hop / rap	22	37	36
Jazz / blues	70	94	71
Rock / pop	114	124	101
Soul / RnB / funk	42	54	46
Unclassied	12	15	4
Total	376	474	320

A correct period estimate is defined to deviate less than 17.5% from the annotated reference and a correct phase to deviate from an annotated beat time less than 0.175 times the annotated period length. This precision requirement has been suggested in [39] and was found perfectly appropriate here since inaccuracies in the manually tapped beat times allow meaningful comparison of only up to that precision. However, for the measure pulse, the period and phase requirements were tightened to 10% and 0.1, respectively, because the measure-period lengths are large and allow the creation of a more accurate reference signal. For the tatum pulse, tactus phase is used and, thus, the phase is correct always when the tactus phase is correct, and only the period has to be considered separately.

Performance rates are given for three different criteria [39].

- Correct: A pulse estimate at time n is accepted if both its period and phase are correct.
- Accept d/h: Consistent period doubling or halving is accepted. More exactly, a pulse estimate is accepted if its phase is correct, the period matches either 0.5, 1.0, or 2.0 times the annotated reference, and the factor does not change within the continuous sequence. Correct meter analysis is taking place but a wrong metrical level is chosen to be, e.g., the tactus pulse.
- Period correct: A pulse estimate is accepted if its period is correct. Phase is ignored. For the tactus pulse, this can be interpreted as the *tempo estimation* accuracy.

Which is the single best number to characterize the performance of a pulse estimator? This was investigated by auralizing

[6]Details of the database can be found online at URL http://www.cs.tut.fi/~klap/iiro/meter.

TABLE II
TACTUS ANALYSIS PERFORMANCE (%) OF DIFFERENT METHODS

	Continuity required			Individual estimates		
Method	Correct	Accept d/h	Period c.	Correct	Accept d/h	Period c.
Causal	57	**68**	74	63	78	76
Noncausal	59	**73**	74	64	80	75
Scheirer [20]	27	**31**	30	48	69	57
Dixon [16]	7	**26**	10	15	53	25
O+Dixon	12	**39**	15	22	63	30

TABLE III
METER ANALYSIS PERFORMANCE OF THE PROPOSED METHOD

		Continuity required			Individual estimates		
Method	Pulse	Correct	Accept d/h	Period	Correct	Accept d/h	Period
Causal	Tatum	44	**57**	62	51	**72**	65
	Tactus	57	**68**	74	63	78	76
	Measure	42	**48**	**78**	43	51	81
Non-causal	Tatum	45	**63**	62	52	**74**	65
	Tactus	59	**73**	74	64	80	75
	Measure	46	**54**	**79**	47	55	81

meter analysis results.[7] It was observed that temporal continuity of correct meter estimates is indeed very important aurally [1, pp. 74,104]. Second, phase errors are very disturbing. Third, period doubling or halving is not very disturbing; tapping *consistently* twice too fast or slow does not matter much and selecting the correct metrical level is in some cases ambiguous even for a human listener [12]. In summary, it appears that the "accept d/h" criterion gives a single best number to characterize the performance of a system.

B. Reference Systems

To put the results in perspective, two reference methods are used as a baseline in simulations. This is essential because the principle of using a continuous sequence of correct estimates for evaluation gives a somewhat pessimistic picture of the absolute performance.

The methods of Scheirer [20] and Dixon [16] are very different, but both systems represent the state-of-the-art in tactus pulse estimation and their source codes are publicly available. Here, the used implementations and parameter values were those of the original authors. However, for Scheirer's method, some parameter tuning was made which slightly improved the results. Dixon developed his system primarily for MIDI-input, and provided only a simple front-end for analyzing acoustic signals. Therefore, a third system denoted "O+Dixon" was developed where an independent onset detector (described in Section II-E) was used prior to Dixon's tactus analysis. Systematic phase errors were compensated for.

C. Experimental Results

In Table II, the tactus tracking performance of the proposed causal and noncausal algorithms is compared with those of the two reference methods. As the first observation, it was noticed that the reference methods did not maintain the temporal continuity of acceptable estimates. For this reason, the performance rates are also given as percentages of individual acceptable estimates (right half of Table II). Dixon's method has difficulties in choosing the correct metrical level for tactus, but performs well according to the "accept d/h" criterion when equipped with the new onset detector. The proposed method outperforms the previous systems in both accuracy and temporal stability.

Table III shows the meter analysis performance of the proposed causal and noncausal algorithms. As for human listeners, meter analysis seems to be easiest at the tactus pulse level. For the measure pulse, period estimation can be done robustly but estimating the phase is difficult. A reason for this is that in a large part of the material, a drum pattern recurs twice within one measure period and the system has difficulties in choosing which one is the first. In the case that π-phase errors (each beat is displaced by a half-period) would be accepted, the performance rate would be essentially the same as for the tactus pulse. However, π-phase errors *are* disturbing and should not be accepted.

For the tatum pulse, in turn, deciding the period is difficult. This is because the temporally atomic pulse rate typically comes up only occasionally, making temporally stable analysis hard to attain. The method often has to halve its period hypothesis when the first rapid event sequence occurs. This appears in the performance rates so that the method is not able to produce a consistent tatum period over time but alternates between, e.g., the reference and double the reference. This degrades the temporally continuous rate, although the "accept d/h" rate is very good for individual estimates. The produced errors are not very disturbing when listening to the results.

As mentioned in Section II-D, the phase analysis of the measure pulse using rhythmic patterns assumes a binary time signature. Nine percent of the pieces in our database have a ternary (3/4) meter but, unfortunately, most of these represent the classical genre where the measure pulse was not annotated. Among the other genres, there were *only five* pieces with ternary meter. For these, the measure-level analysis was approximately twice less accurate than for the rest of the database. For the tactus and tatum, there were 41 and 30 annotated ternary pieces, respectively, and no significant degradation in performance was observed. On the contrary, the ternary pieces were rhythmically easier than the others within the same genre.

Fig. 10 shows the "accept d/h" (continuity required) performance rates for the proposed causal system within different musical genres. For classical music, the proposed method is only moderately successful, although, e.g., the tactus estimation error rate still outperforms the performance of the reference methods for the whole material (31% and 26% for Scheirer's and Dixon's methods, respectively). However, this may suggest that pitch analysis would be needed to analyze the meter of classical music. In jazz music, the complexity of musical rhythms is higher on the average and the task, thus, harder.

D. Importance of the Different Parts of the Probability Model

Table IV shows the performance rates for different system configurations. Different elements of the proposed model were disabled in order to evaluate their importance. In each case, the system was kept otherwise fixed. The baseline method is the noncausal system.

[7]Samples are available at URL http://www.cs.tut.fi/~klap/iiro/meter.

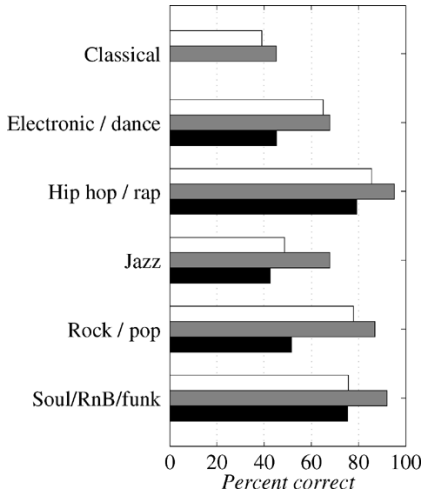


Fig. 10. Performance of the proposed causal system within different musical genres. The "accept d/h" (continuity required) percentages are shown for the tatum (white), tactus (gray), and measure pulses (black).

TABLE IV
METER ANALYSIS PERFORMANCE (%) FOR DIFFERENT SYSTEM CONFIGURATIONS

	Continuity required, accept d/h			Individual estimates, accept d/h		
Method	Tatum	Tactus	Measure	Tatum	Tactus	Measure
0. Baseline	**63**	**73**	**54**	74	80	55
1. No joint estim.	**58**	**68**	**49**	71	75	50
2. No temporal proc.	**45**	**54**	**31**	72	77	50
3. Neither of the two	**41**	**50**	**25**	70	72	44

In the first test, the dependencies between the different pulse levels were broken by using a noninformative (flat) distribution for $g(x)$ in (25). This slightly degrades the performance in all cases. In the second test, the dependencies between temporally successive estimates were broken by using a noninformative distribution for the transition probabilities between successive period and phase estimates, $P(\tau_n^i \mid \tau_{n-1}^i)$ and $P(\varphi_n^i \mid \varphi_{n-1}^i)$, respectively. This degrades the temporal stability of the estimates considerably and, hence, collapses the performance rates which use the longest continuous correct segment for evaluation. In the third case, the both types of dependencies were broken. The system still performs moderately, indicating that the initial time-frequency analysis method and the comb-filter resonators provide a high level of robustness.

IV. CONCLUSION

A method has been described which can successfully analyze the meter of acoustic musical signals. Musical genres of very diverse types can be processed with a common system configuration and parameter values. For most musical material, relatively low-level acoustic information can be used, without the need to model the higher level auditory functions such as sound source separation or multipitch analysis.

Similarly to human listeners, computational meter analysis is easiest at the tactus pulse level. For the measure pulse, period estimation can be done equally robustly but estimating the phase is less straightforward. Either rhythmic pattern matching or pitch analysis seems to be needed to analyze music at this time scale. For the tatum pulse, in turn, phase estimation is not difficult at all, but deciding the period is very difficult for both humans and a computational algorithm. This is because the temporally atomic pulse rate typically comes up only occasionally. Thus, causal processing is difficult and it is often necessary to halve the tatum hypothesis when the first rapid event sequence occurs.

The critical elements of a meter analysis system appear to be the initial time-frequency analysis part which measures musical accentuation as a function of time and the (often implicit) internal model which represents primitive musical knowledge. The former is needed to provide robustness for diverse instrumentations in classical, rock, or electronic music, for example. The latter is needed to achieve temporally stable meter tracking and to fill in parts where the meter is only faintly implied by the musical surface. A challenge in this part is to develop a model which is generic for jazz and classical music, for example. The proposed model describes sufficiently low-level musical knowledge to generalize over different genres.

APPENDIX

This appendix presents the derivation and underlying assumptions in the estimation of the state-conditional observation likelihoods $p(\boldsymbol{s} \mid \boldsymbol{q})$. We first assume that the realizations of τ^{A} are independent of the realizations of τ^{B} and τ^{C}, that is, $P(\tau^{\mathrm{A}} = j \mid \tau^{\mathrm{B}} = k, \tau^{\mathrm{C}} = l) = P(\tau^{\mathrm{A}} = j)$. This violates the dependencies of our model but significantly simplifies the computations and makes it possible to obtain reasonable estimates. Using the assumption, we can write

$$P(\boldsymbol{s} \mid \tau^{\mathrm{A}} = j, \tau^{\mathrm{B}} = k, \tau^{\mathrm{C}} = l) = \frac{P(\boldsymbol{s} \mid \tau^{\mathrm{B}} = k, \tau^{\mathrm{C}} = l) P(\boldsymbol{s} \mid \tau^{\mathrm{A}} = j)}{P(\boldsymbol{s})}. \quad (35)$$

Furthermore, tatum information is most clearly visible in the spectrum of the resonator outputs. Thus, we use

$$P(\boldsymbol{s} \mid \tau^{\mathrm{A}} = j) = P(\boldsymbol{S} \mid \tau^{\mathrm{A}} = j) \quad (36)$$

where $\boldsymbol{S}$ is the spectrum of $\boldsymbol{s}$, according to (10). We further assume the components of $\boldsymbol{s}$ and $\boldsymbol{S}$ to be conditionally independent of each other given the state, and write the nominator of (35) as

$$P(\boldsymbol{s} \mid \tau^{\mathrm{B}} = k, \tau^{\mathrm{C}} = l) P(\boldsymbol{S} \mid \tau^{\mathrm{A}} = j) = \prod_{k'=1}^{\tau_{\max}} P(s(k') \mid \tau^{\mathrm{B}} = k, \tau^{\mathrm{C}} = l) \prod_{j'=1}^{\tau_{\max}} P\left(S\left(\frac{1}{j}'\right) \mid \tau^{\mathrm{A}} = j\right). \quad (37)$$

We make two more simplifying assumptions. First, we assume that the value of $\boldsymbol{s}$ and $\boldsymbol{S}$ at the lags corresponding to a period actually present in the signal depends only on the particular period, not on other periods. Second, the value at lags where there is no period present in the signal is independent of the true periods τ^{A}, τ^{B}, and τ^{C}, and is dominated by the fact

that no period corresponds to that particular lag. Hence, (35) can be written as

$$\begin{aligned} &P(\boldsymbol{s} \mid \boldsymbol{q} = [j,k,l]) \\ &= \frac{1}{P(\boldsymbol{s})} P(s(k) \mid \tau^{\mathrm{B}} = k) \\ &\quad \cdot P(s(l) \mid \tau^{\mathrm{C}} = l) \prod_{k' \neq k, \boldsymbol{l}} P(s(k') \mid \tau^{\mathrm{B}}, \tau^{\mathrm{C}} \neq k') \\ &\quad \cdot P\left(S\left(\frac{1}{j}\right) \mid \tau^{\mathrm{A}} = j\right) \prod_{j' \neq j} P\left(S\left(\frac{1}{j'}\right) \mid \tau^{\mathrm{A}} \neq j'\right) \end{aligned} \quad (38)$$

where $P(s(\tau) \mid \tau^{\mathrm{B}} = \tau)$ denotes the probability of value $s(\tau)$ given that τ is a tactus pulse period and $P(s(\tau) \mid \tau^{\mathrm{B}} \neq \tau)$ denotes the probability of value $s(\tau)$ given that τ is not a tactus pulse period. These conditional probability distributions (tactus, measure, and tatum each have two distributions) were approximated by discretizing the value range of $s(\tau)$, $s(\tau) \in [0,1]$, and by calculating a histogram of $s(\tau)$ values in the cases that τ is or is not an annotated metrical pulse period.

Then, by defining

$$\begin{aligned} \beta(\boldsymbol{s}) = \frac{1}{P(\boldsymbol{s})} &\prod_{k'=1}^{\tau_{\max}} P(s(k') \mid \tau^{\mathrm{B}}, \tau^{\mathrm{C}} \neq k') \\ &\cdot \prod_{j'=1}^{\tau_{\max}} P\left(S\left(\frac{1}{j'}\right) \mid \tau^{\mathrm{A}} \neq j'\right) \end{aligned} \quad (39)$$

Equation (38) can be written as

$$\begin{aligned} P(\boldsymbol{s} \mid \boldsymbol{q} = [j,k,l]) = \beta(\boldsymbol{s}) &\cdot \frac{P(s(k) \mid \tau^{\mathrm{B}} = k)}{P(s(k) \mid \tau^{\mathrm{B}}, \tau^{\mathrm{C}} \neq k)} \\ &\cdot \frac{P(s(l) \mid \tau^{\mathrm{C}} = l)}{P(s(l) \mid \tau^{\mathrm{B}}, \tau^{\mathrm{C}} \neq l)} \frac{P\left(S\left(\frac{1}{j}\right) \mid \tau^{\mathrm{A}} = j\right)}{P\left(S\left(\frac{1}{j}\right) \mid \tau^{\mathrm{A}} \neq j\right)} \end{aligned} \quad (40)$$

where the scalar $\beta(\boldsymbol{s})$ is a function of $\boldsymbol{s}$ but does not depend on $\boldsymbol{q}$.

By using the two approximated histograms for the tactus, measure, and tatum pulses, each of the three terms of the form $P(s(\tau) \mid \tau^i = \tau)/P(s(\tau) \mid \tau^i \neq \tau)$ in (40) can be represented by a single discrete histogram. These were modeled with first-order polynomials. The first two terms depend linearly on the value $s(\tau)$ and the last term depends linearly on the value $S(1/\tau)$. Thus, we can write

$$p(\boldsymbol{s} \mid \boldsymbol{q} = [j,k,l]) \propto s(k)s(l)S\left(\frac{1}{j}\right). \quad (41)$$

The histograms could be more accurately modeled with third-order polynomials, but this did not bring performance advantage over the simple linear model in (41).

Numerical values of the matrices used in Section II-D

$$\eta_{c,k}^{(1)} = \begin{bmatrix} 12 & 1.0 & 0 & 5.7 \\ 0 & 2.0 & 0 & 2.0 \\ 0 & 3.0 & 0 & 3.0 \\ 0 & 4.0 & 0 & 4.0 \end{bmatrix}, \quad \eta_{c,k}^{(2)} = \begin{bmatrix} 10 & 0 & 1.4 & 1.3 \\ 0 & 0 & 2.8 & 0.8 \\ 0 & 0 & 4.3 & 1.2 \\ 0 & 0 & 5.8 & 1.5 \end{bmatrix}.$$

Here channel c determines the row and delay k the column. The first row corresponds to the lowest-frequency channel.

REFERENCES

[1] F. Lerdahl and R. Jackendoff, *A Generative Theory of Tonal Music*. Cambridge, MA: MIT Press, 1983.

[2] E. F. Clarke, "Rhythm and timing in music," in *The Psychology of Music*, D. Deutsch, Ed. New York: Academic, 1999.

[3] J. Bilmes, "Timing is of the essence: perceptual and computational techniques for representing, learning, and reproducing expressive timing in percussive rhythm," Master's thesis, School of Architecture and Planning, Massachusetts Inst. Technol., Cambridge, MA, 1993.

[4] F. Gouyon, L. Fabig, and J. Bonada, "Rhythmic expressiveness transformations of audio recordings: Swing modifications," in *Proc. 6th Int. Conf. Digital Audio Effects*, London, U.K., 2003, pp. 94–99.

[5] A. T. Cemgil and B. Kappen, "Monte Carlo methods for tempo tracking and rhythm quantization," *J. Artif. Intell. Res.*, vol. 18, pp. 45–81, 2003.

[6] A. P. Klapuri, "Multiple fundamental frequency estimation based on harmonicity and spectral smoothness," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 6, pp. 804–815, Nov. 2003.

[7] J. K. Paulus and A. P. Klapuri, "Conventional and periodic N-grams in the transcription of drum sequences," in *Proc. IEEE Int. Conf. Multimedia and Expo.*, Baltimore, MD, Jul. 2003, pp. 737–740.

[8] M. Ryynänen and A. P. Klapuri, "Modeling of note events for singing transcription," in *Proc. ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio*, Jeju, Korea, Oct. 2004.

[9] C. S. Lee, "The perception of metrical structure: Experimental evidence and a model," in *Representing Musical Structure*, P. Howell, R. West, and I. Cross, Eds. New York: Academic, 1991.

[10] P. Desain and H. Honing, "Computational models of beat induction: The rule-based approach," *J. New Music Res.*, vol. 28, no. 1, pp. 29–42, 1999.

[11] D. F. Rosenthal, "Machine rhythm: Computer emulation of human rhythm perception," Ph.D. dissertation, MIT Media Lab., Mass. Inst. Technol., Cambridge, 1992.

[12] R. Parncutt, "A perceptual model of pulse salience and metrical accent in musical rhythms," *Music Percept.*, vol. 11, no. 4, pp. 409–464, 1994.

[13] J. C. Brown, "Determination of the meter of musical scores by autocorrelation," *J. Acoust. Soc. Amer.*, vol. 94, no. 4, pp. 1953–1957, 1993.

[14] E. W. Large and J. F. Kolen, "Resonance and the perception of musical meter," *Connection Sci.*, vol. 6, no. 1, pp. 177–208, 1994.

[15] D. Temperley and D. Sleator, "Modeling meter and harmony: A preference-rule approach," *Comput. Music J.*, vol. 23, no. 1, pp. 10–27, 1999.

[16] S. Dixon, "Automatic extraction of tempo and beat from expressive performances," *J. New Music Res.*, vol. 30, no. 1, pp. 39–58, 2001.

[17] C. Raphael, "Automated rhythm transcription," in *Proc. Int. Symp. Music Information Retrieval*, Oct. 2001, pp. 99–107.

[18] M. Goto and Y. Muraoka, "Music understanding at the beat level — Real-time beat tracking for audio signals," in *Proc. IJCAI-95 Workshop on Computational Auditory Scene Analysis*, 1995, pp. 68–75.

[19] ——, "Real-time rhythm tracking for drumless audio signals — Chord change detection for musical decisions," in *Proc. IJCAI-97 Workshop on Computational Auditory Scene Analysis*, 1997, pp. 135–144.

[20] E. D. Scheirer, "Tempo and beat analysis of acoustic musical signals," *J. Acoust. Soc. Amer.*, vol. 103, no. 1, pp. 588–601, 1998.

[21] W. A. Sethares and T. W. Staley, "Meter and periodicity in musical performance," *J. New Music Res.*, vol. 22, no. 5, 2001.

[22] J. Laroche, "Efficient tempo and beat tracking in audio recordings," *J. Audio Eng. Soc.*, vol. 51, no. 4, pp. 226–233, Apr. 2003.

[23] S. Hainsworth and M. Macleod, "Beat tracking with particle filtering algorithms," in *Proc. IEEE Workshop on Applications of Signal Proc. to Audio and Acoustics*, New Paltz, NY, 2003, pp. 91–94.

[24] F. Gouyon, P. Herrera, and P. Cano, "Pulse-dependent analyzes of percussive music," in *Proc. AES 22nd Int. Conf. Virtual, Synthetic and Entertainment Audio*, Espoo, Finland, Jun. 2002, pp. 396–401.

[25] A. P. Klapuri, "Sound onset detection by applying psychoacoustic knowledge," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Phoenix, AZ, 1999, pp. 3089–3092.
[26] D. Moelants and C. Rampazzo, "A computer system for the automatic detection of perceptual onsets in a musical signal," in *KANSEI — The Technology of Emotion*, A. Camurri, Ed. Genova, Switzerland: AIMI/DIST, 1997, pp. 141–146.
[27] M. Davy and S. Godsill, "Detection of abrupt spectral changes using support vector machines. An application to audio signal segmentation," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Orlando, FL, May 2002, pp. 1313–1316.
[28] S. A. Abdallah and M. D. Plumbley, "Probability as metadata: Event detection in music using ICA as a conditional density model," in *Proc. 4th Int. Symp. Independent Component Analysis and Blind Signal Separation*, Nara, Japan, Apr. 2003, pp. 233–238.
[29] C. Duxbury, J. P. Bello, M. Davies, and M. Sandler, "Complex domain onset detection for musical signals," in *Proc. 6th Int. Conf. on Digital Audio Effects*, London, U.K., Sep. 2003.
[30] R. Meddis and M. J. Hewitt, "Virtual pitch and phase sensitivity of a computer model of the auditory periphery. I: Pitch identification," *J. Acoust. Soc. Amer.*, vol. 89, no. 6, pp. 2866–2882, 1991.
[31] *Hearing. Handbook of Perception and Cognition*, 2nd ed., B. C. J. Moore, Ed., Academic, New York, 1995.
[32] R. Meddis, "Simulation of mechanical to neural transduction in the auditory receptor," *J. Acoust. Soc. Amer.*, vol. 79, no. 3, pp. 702–711, 1986.
[33] A. de Cheveigné and H. Kawahara, "YIN, A fundamental frequency estimator for speech and music," *J. Acoust. Soc. Amer.*, vol. 111, no. 4, pp. 1917–1930, 2002.
[34] R. C. Maher and J. W. Beauchamp, "Fundamental frequency estimation of musical signals using a two-way mismatch procedure," *J. Acoust. Soc. Amer.*, vol. 95, no. 4, pp. 2254–2263, 1994.
[35] L. van Noorden and D. Moelants, "Resonance in the perception of musical pulse," *J. New Music Res.*, vol. 28, no. 1, pp. 43–66, 1999.
[36] J. Seppänen, "Computational models of musical meter recognition," Master's thesis, Dept. Inf. Technol., Tampere Univ. Technol., Tampere, Finland, 2001.
[37] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Stat. Soc.*, vol. 39, pp. 1–38, 1977.
[38] L. R. Rabiner and B.-H Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Prentice-Hall, 1993.
[39] M. Goto and Y. Muraoka, "Issues in evaluating beat tracking systems," in *Proc. IJCAI-97 Workshop on Issues in AI and Music*, 1997, pp. 9–16.

Anssi P. Klapuri was born in Kälviä, Finland, in 1973. He received the M.Sc. and Dr.Tech. degrees in information technology from the Tampere University of Technology (TUT), Tampere, Finland, in 1998 and 2004, respectively.

He has been with the TUT Institute of Signal Processing since 1996. His research interests include automatic transcription of music and audio signal processing in general.

Antti Eronen was born in Ilomantsi, Finland, in 1977. He received the M.S. degree in information technology from Tampere University of Technology (TUT), Tampere, Finland, in 2001. He is currently pursuing a postgraduate degree.

From 1998 to 2003, he was with the Institute of Signal Processing of TUT. In 2003, he joined Nokia Research Center, Tampere, Finland. His research interests include content recognition, analysis, and synthesis of music and audio signals.

Jaakko Astola (F'00) received the B.Sc., M.Sc., Licentiate, and Ph.D. degrees in mathematics (specializing in error-correcting codes) from Turku University, Finland, in 1972, 1973, 1975, and 1978 respectively.

From 1976 to 1977, he was with the Research Institute for Mathematical Sciences, Kyoto University, Kyoto, Japan. From 1979 to 1987, he was with the Department of Information Technology, Lappeenranta University of Technology, Lappeenranta, Finland, holding various teaching positions in mathematics, applied mathematics and computer science. In 1984, he worked as a Visiting Scientist at the Eindhoven University of Technology, Eindhoven, The Netherlands. From 1987 to 1992, he was an Associate Professor in Applied Mathematics at Tampere University, Tampere, Finland. Since 1993, he has been a Professor of Signal Processing and Director of Tampere International Center for Signal Processing leading a group of about 60 scientists and was nominated Academy Professor by Academy of Finland (2001–2006). His research interests include signal processing, coding theory, spectral techniques and statistics.

Publication 6

J. Seppänen, A. Eronen, J. Hiipakka, "Joint Beat & Tatum Tracking from Music Signals", *Proceedings of the 7th International Conference on Music Information Retrieval, ISMIR 2006*, Victoria, Canada, October 2006.

Joint Beat & Tatum Tracking from Music Signals

Jarno Seppänen Antti Eronen Jarmo Hiipakka
Nokia Research Center
P.O.Box 407, FI-00045 NOKIA GROUP, Finland
{jarno.seppanen, antti.eronen, jarmo.hiipakka}@nokia.com

Abstract

This paper presents a method for extracting two key metrical properties, the beat and the tatum, from acoustic signals of popular music. The method is computationally very efficient while performing comparably to earlier methods. High efficiency is achieved through multirate accent analysis, discrete cosine transform periodicity analysis, and phase estimation by adaptive comb filtering. During analysis, the music signals are first represented in terms of accentuation on four frequency subbands, and then the accent signals are transformed into periodicity domain. Beat and tatum periods and phases are estimated in a probabilistic setting, incorporating primitive musicological knowledge of beat–tatum relations, the prior distributions, and the temporal continuities of beats and tatums. In an evaluation with 192 songs, the beat tracking accuracy of the proposed method was found comparable to the state of the art. Complexity evaluation showed that the computational cost is less than 1% of earlier methods. The authors have written a real-time implementation of the method for the S60 smartphone platform.

Keywords: Beat tracking, music meter estimation, rhythm analysis.

1. Introduction

Recent years have brought significant advances in the field of automatic music signal analysis, and music meter estimation is no exception. In general, the music meter contains a nested grouping of pulses called *metrical levels*, where pulses on higher levels are subsets of the lower level pulses; the most salient level is known as the *beat*, and the lowest level is termed the *tatum* [1, p. 21].

Metrical analysis of music signals has many applications ranging from browsing and visualization to classification and recommendation of music. The state of the art has advanced high in performance, but the computational requirements have also remained restrictively high. The proposed method significantly improves computational efficiency while maintaining satisfactory performance.

The technical approaches for meter estimation are various, including *e.g.* autocorrelation based methods [6], inter-onset interval histogramming [5], or banks of comb filter resonators [4], possibly followed by a probabilistic model [3]. See [2] for a review on rhythm analysis systems.

2. Algorithm Description

The algorithm overview is presented in Fig. 1: the input is audio signals of polyphonic music, and the output consists of the times of beats and tatums. The implementation of the beat and tatum tracker has been done in C++ programming language in the S60 smartphone platform. The algorithm design is causal and the implementation works in real time.

The operation of the system can be described in six stages (see Fig. 1):

1. Resampling stage,
2. Accent filter bank stage,
3. Buffering stage,
4. Periodicity estimation stage,
5. Period estimation stage, and
6. Phase estimation stage.

First, the signal is resampled to a fixed sample rate, to support arbitrary input sample rates. Second, the accent filter bank transforms the acoustic signal of music into a form that is suitable for beat and tatum analysis. In this stage, subband accent signals are generated, which constitute an estimate of the perceived accentuation on each subband. The accent filter bank stage significantly reduces the amount of data.

Then, the accent signals are accumulated into four-second frames. Periodicity estimation looks for repeating accents on each subband. The subband periodicities are then combined, and summary periodicity is computed.

Next, the most likely beat and tatum periods are estimated from each periodicity frame. This uses a probabilistic formulation of primitive musicological knowledge, including the relation, the prior distribution, and the temporal continuity of beats and tatums. Finally, the beat phase is found and beat and tatum times are positioned. The accent signal is filtered with a pair of comb filters, which adapt to different beat period estimates.

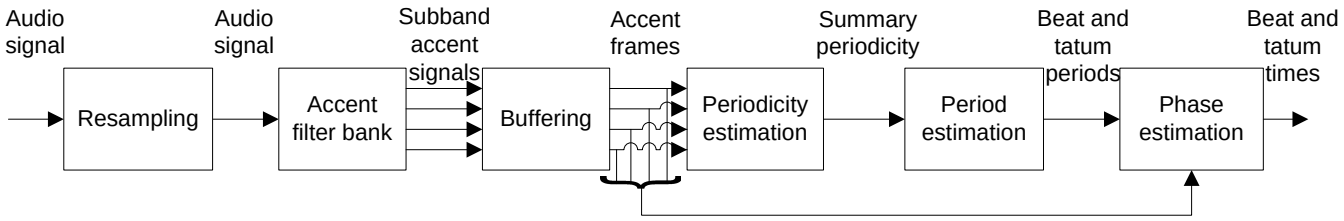


Figure 1. Beat and tatum analyzer.

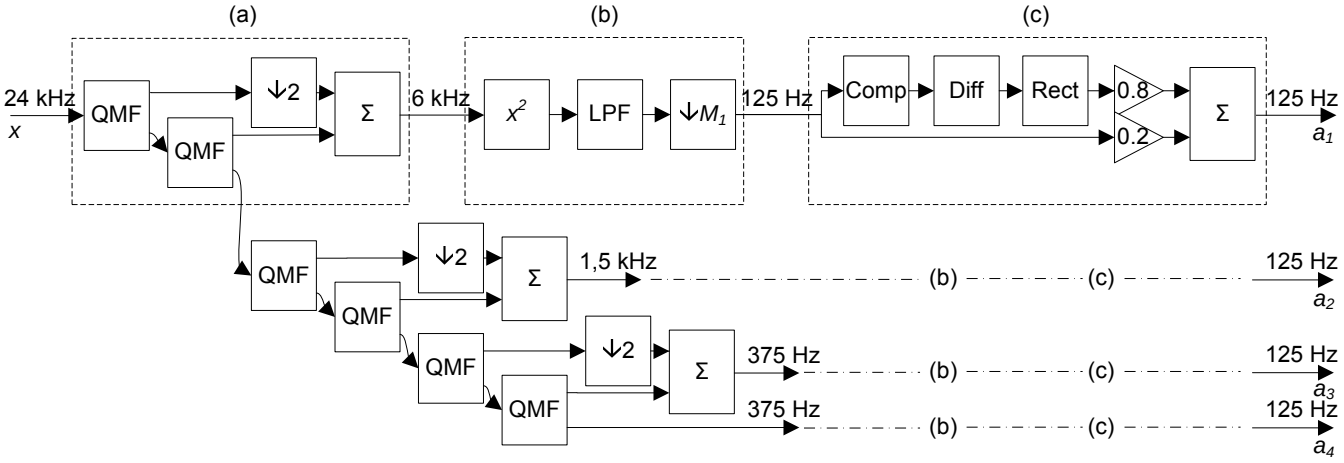


Figure 2. Accent filter bank overview. (a) The audio signal is first divided into subbands, then (b) power estimates on each subband are calculated, and (c) accent computation is performed on the subband power signals.

2.1. Resampling

Before any audio analysis takes place, the signal is converted to a 24 kHz sample rate. This is required because the filter bank uses fixed frequency regions. The resampling can be done with a relatively low-quality algorithm, linear interpolation, because high fidelity is not required for successful beat and tatum analysis.

2.2. Accent Filter Bank

Figure 2 presents an overview of the accent filter bank. The incoming audio signal $x[n]$ is (a) first divided into subband audio signals, and (b) a power estimate signal is calculated for each band separately. Last, (c) an accent signal is computed for each subband.

The filter bank divides the acoustic signal into seven frequency bands by means of six cascaded decimating quadrature mirror filters (QMF). The QMF subband signals are combined pairwise into three two-octave subband signals, as shown in Fig. 2(a). When combining two consecutive branches, the signal from the higher branch is decimated without filtering. However, the error caused by the aliasing produced in this operation is negligible for the proposed method. The sampling rate decreases by four between successive bands due to the two QMF analysis stages and the extra decimation step. As a result, the frequency bands are located at 0–190 Hz, 190–750 Hz, 750–3000 Hz, and 3–12 kHz, when the filter bank input is at 24 kHz.

There is a very efficient structure that can be used to implement the downsampling QMF analysis with just two allpass filters, an addition, and a subtraction. This structure is depicted in Fig. 5.2-5 in [7, p. 203]. The allpass filters for this application can be first-order filters, because only modest separation is required between bands.

The subband power computation is shown Fig. 2(b). The audio signal is squared, low-pass filtered (LPF), and decimated by subband specific factor M_i to get the subband power signal. The low-pass filter is a digital filter having 10 Hz cutoff frequency. The subband decimation ratios $M_i = \{48, 12, 3, 3\}$ have been chosen so that the power signal sample rate is 125 Hz on all subbands.

The subband accent signal computation in Fig. 2(c) is modelled according to Klapuri *et al.* [3, p. 344–345]. In the process, the power signal first is mapped with a nonlinear level compression function labeled *Comp* in Fig. 2(c),

$$f(x) = \begin{cases} 5.213 \ln(1 + 10\sqrt{x}), & x > 0.0001 \\ 5.213 \ln 1.1 & \text{otherwise.} \end{cases} \quad (1)$$

Following compression, the first-order difference signal is computed (*Diff*) and half-wave rectified (*Rect*). In accordance with Eq. (3) in [3], the rectified signal is summed to the power signal after constant weighting, see Fig. 2(c). The high computational efficiency of the proposed method lies mostly in the accent filter bank design. In addition to efficiency, the resulting accent signals are comparable to those of Klapuri *et al.*, see *e.g.* Fig. 3 in [3].

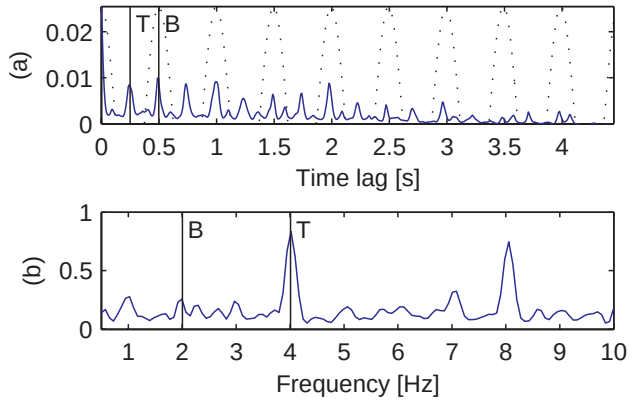


Figure 3. (a) Normalized autocorrelation and (b) summary periodicity, with beat (B) and tatum (T) periods shown.

2.3. Buffering

The buffering stage implements a ring buffer which accumulates the signal into fixed-length frames. The incoming signal is split into consecutive accent signal frames of a fixed length $N = 512$ (4.1 seconds). The value of N can be modified to choose a different performance–latency tradeoff.

2.4. Accent Periodicity Estimation

The accent signals are analyzed for intrinsic repetitions. Here, periodicity is defined as the combined strength of accents that repeat with a given period. For all subband accent signals, a joint summary periodicity vector is computed.

Autocorrelation $\rho[\ell] = \sum_{n=0}^{N-1} a[n]a[n-\ell]$, $0 \leq \ell \leq N-1$, is first computed from each N-length subband accent frame $a[n]$. The accent signal reaches peak values whenever there are high accents in the music and remains low otherwise. Computing autocorrelation from an impulsive accent signal is comparable to computing the inter-onset interval (IOI) histogram as described by Seppänen [5], with additional robustness due to not having to discretize the accent signal into onsets.

The accent frame power $\rho[0]$ is stored for later weighting of subband periodicities. Offset and scale variations are eliminated from autocorrelation frames by normalization,

$$\bar{\rho}[\ell] = \frac{\rho[\ell] - \min_n \rho[n]}{\sum_{n=0}^{N-1} \rho[n] - N \min_n \rho[n]}. \qquad (2)$$

See Fig. 3(a) for an example normalized autocorrelation frame. The figure shows also the correct beat period B, 0.5 seconds, and tatum period T, 0.25 seconds, as vertical lines.

Next, accent periodicity is estimated by means of the N-point discrete cosine transform (DCT)

$$R[k] = c_k \sum_{n=0}^{N-1} \bar{\rho}[n] \cos \frac{\pi(2n+1)k}{2N} \qquad (3)$$

$$c_0 = \sqrt{1/N} \qquad (4)$$

$$c_k = \sqrt{2/N}, \qquad 1 \leq k \leq N-1. \qquad (5)$$

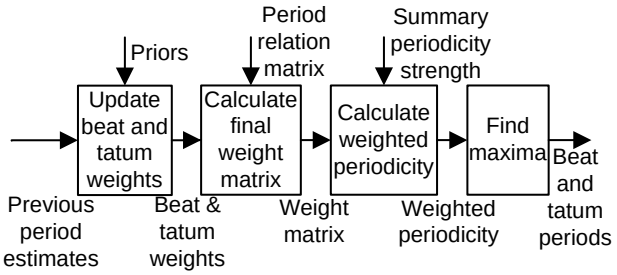


Figure 4. The period estimator.

Similarly to an IOI histogram [5], accent peaks with a period p cause high responses in the autocorrelation function at lags $\ell = 0$, $\ell = p$ (nearest peaks), $\ell = 2p$ (second-nearest peaks), $\ell = 3p$ (third-nearest peaks), and so on. Such response is exploited in DCT-based periodicity estimation, which matches the autocorrelation response with zero-phase cosine functions; see dashed lines in Fig. 3(a).

Only a specific periodicity window, $0.1 \text{ s} \leq p \leq 2 \text{ s}$, is utilized from the DCT vector $R[k]$. This window specifies the range of beat and tatum periods for estimation. The subband periodicities $R_i[k]$ are combined into an M-point summary periodicity vector, $M = 128$,

$$S[k] = \sum_{i=1}^{4} \rho_i[0]^\gamma \widetilde{R}_i[k] \quad 0 \leq k \leq M-1, \qquad (6)$$

where $\widetilde{R}_i[k]$ has interpolated values of $R_i[k]$ from 0.5 Hz to 10 Hz, and the parameter $\gamma = 1.2$ controls weighting. Figure 3(b) shows an example summary periodicity vector.

2.5. Beat and Tatum Period Estimation

The period estimation stage finds the most likely beat period $\hat{\tau}_n^B$ and tatum period $\hat{\tau}_n^A$ for the current frame at time n based on the observed periodicity $S[k]$ and primitive musicological knowledge. Likelihood functions are used for modeling primitive musicological knowledge as proposed by Klapuri *et al.* in [3, p. 344–345], although the actual calculations of the model are different. An overview of the period estimator are depicted in Fig. 4.

First, weights $f^i(\tau_n^i)$ for the different beat and tatum period candidates are calculated as a product of prior distributions $p^i(\tau^i)$ and "continuity functions":

$$f_C^i\left(\frac{\tau_n^i}{\tau_{n-1}^i}\right) = \frac{1}{\sigma_1\sqrt{2\pi}} \exp\left[-\frac{1}{2\sigma_1^2}\left(\ln \frac{\tau_n^i}{\tau_{n-1}^i}\right)^2\right], \qquad (7)$$

as defined in Eq. (21) in [3, p. 348]. Here, $i = A$ denotes the tatum and $i = B$ denotes the beat. The value $\sigma_1 = 0.63$ is used. The continuity function describes the tendency that the periods are slowly varying, thus taking care of "tying" the successive period estimates together. τ_{n-1}^i is defined as the median of three previous period estimates. This is found to be slightly more robust than just using the estimate from

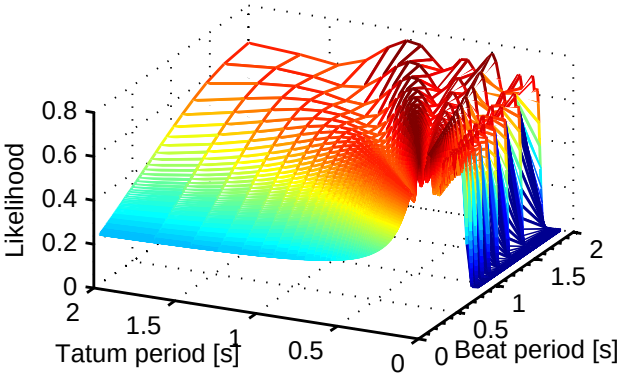


Figure 5. Likelihood of different beat and tatum periods to occur jointly.

the previous frame. The priors are lognormal distributions as described in Eq. (22) in [3, p. 348].

The output of the *Update beat and tatum weights* step in Fig. 4 are two weighting vectors containing the evaluated values of the functions $f^B(\tau_n^B)$ and $f^A(\tau_n^A)$. The values are obtained by evaluating the continuity functions for the set of possible periods given the previous beat and tatum estimates, and multiplying with the priors.

The next step, *Calculate final weight matrix*, adds in the modelling of the most likely relations between simultaneous beat and tatum periods. For example, the beat and tatum are more likely to occur at ratios of 2, 4, 6, and 8 than in ratios of 1, 3, 5, and 7. The likelihood of possible beat and tatum period combinations τ^B, τ^A is modelled with a Gaussian mixture density, as described in Eq. (25) in [3, p. 348]:

$$g(\tau^B, \tau^A) = \sum_{l=1}^{9} w_l \mathcal{N}(\frac{\tau^B}{\tau^A}; l, \sigma_2) \quad (8)$$

where l are the component means and σ_2 is the common variance. Eq. (8) is evaluated for the set of $M \times M$ period combinations. The weights w_l were hand adjusted to give good performance on a small set of test data. Fig. 5 depicts the resulting likelihood surface $g(\tau^B, \tau^A)$. The final weighting function is

$$h(\tau_n^B, \tau_n^A) = \sqrt{f^B(\tau_n^B)\sqrt{g(\tau_n^B, \tau_n^A) f^A(\tau_n^A)}}. \quad (9)$$

Taking the square root spreads the function such that the peaks do not become too narrow. The result is a final $M \times M$ likelihood weighting matrix $\mathbf{H}$ with values of $h(\tau_n^B, \tau_n^A)$ for all beat and tatum period combinations.

The *Calculate weighted periodicity* step weights the summary periodicity observation with the obtained likelihood weighting matrix $\mathbf{H}$. We assume that the likelihood of observing a certain beat and tatum combination is proportional to a sum of the corresponding values of the summary periodicity, and define the observation $O(\tau_n^B, \tau_n^A) = (S[k_B] + S[k_A])/2$, where the indices k_B and k_A correspond to the periods τ_n^B and τ_n^A, respectively. This gives an observation matrix of the same size as our weighting matrix. The observation matrix is multiplied pointwise with the weighting matrix, giving the weighted $M \times M$ periodicity matrix $\mathbf{P}$ with values $P(\tau_n^B, \tau_n^A) = h(\tau_n^B, \tau_n^A) O(\tau_n^B, \tau_n^A)$. The final step is to *Find the maximum* from $\mathbf{P}$. The indices of the maximum correspond to the beat and tatum period estimates $\hat{\tau}_n^B$, $\hat{\tau}_n^A$. The period estimates are passed on to the phase estimator stage.

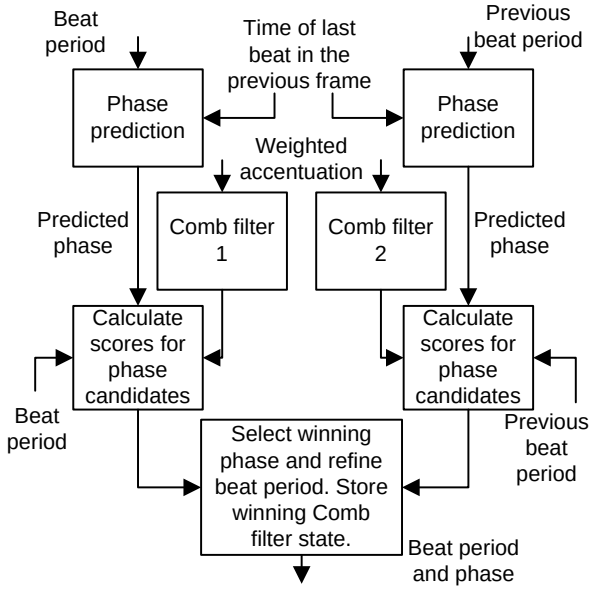


Figure 6. The phase estimation stage finds the phase of the beat and tatum pulses, and may also refine the beat period estimate.

2.6. Beat Phase Estimation

The phase estimation stage is depicted in Fig. 6. The tatum phase is the same as the beat phase and, thus, only the beat phase is estimated. Phase estimation is based on a weighted sum $v[n] = \sum_{i=1}^{4}(6-i)a_i[n]$ of the observed subband accent signals $a_i[n]$, $0 \leq n \leq N-1$. Compared to Eq. (27) in [3, p. 350], the summation is done directly across the accent subbands, instead of resonator outputs.

A bank of comb filters with constant half time T_0 and delays corresponding to different period candidates have been found to be a robust way of measuring the periodicity in accentuation signals [3] [4]. Another benefit of comb filters is that an estimate of the phase of the beat pulse is readily obtained by examining the comb filter states [4, p. 593]. However, implementing a bank of comb filters across the range of possible beat and tatum periods is computationally very expensive. The proposed method utilizes the benefits of comb filters with a fraction of the computational cost of the earlier methods. The phase estimator implements two comb filters. The output of a comb filter with delay τ and

gain α_τ for the input $v[n]$ is given by

$$r[n] = \alpha_\tau r[n-\tau] + (1-\alpha_\tau)v[n]. \quad (10)$$

The parameter τ of the two comb filters is continuously adapted to match the current ($\hat{\tau}_n^B$) and the previous ($\hat{\tau}_{n-1}^B$) period estimates. The feedback gain $\alpha_\tau = 0.5^{\tau/T_0}$, where the half time T_0 corresponds to three seconds in samples.

The phase estimation starts by finding a prediction $\hat{\phi}_n$ for the beat phase ϕ_n in this frame, the step *Phase prediction* in Fig. 6. The prediction is calculated by adding the current beat period estimate to the time of the last beat in the previous frame. Another source of phase prediction is the comb filter state, however, this is not always available since the filter states may be reset between frames.

The accent signal is passed through the Comb filter 1, giving the output $r_1[n]$. If there are peaks in the accent signal corresponding to the comb filter delay, the output level of the comb filter will be large due to a resonance.

We then calculate a score for the different phase candidates $l = 0, \ldots, \hat{\tau}_B - 1$ in this frame. The score is

$$p[l] = \frac{1}{|I_l|} \sum_{j \in I_l} r_1[j] \quad (11)$$

where I_l is the set of indices $\{l, l+\hat{\tau}_B, l+2\hat{\tau}_B, \ldots\}$ belonging to the current frame, $\forall i \in I_l : 0 \leq i \leq N-1$. The scores are weighted by a function which depends on the deviation of the phase candidate from the predicted phase value. More precisely, the weight is calculated according to Eq. (33) in [3, p. 350]:

$$w[l] = \frac{1}{\sigma_3\sqrt{2\pi}} \exp\left(-\frac{d[l]^2}{2\sigma_3^2}\right), \quad (12)$$

but the distance is calculated in a simpler way: $d[l] = (l - \hat{\phi}_n)/\hat{\tau}_B$. The phase estimate is the value of l maximizing $p[l]w[l]$.

If there are at least three beat period predictions available and the beat period estimate has changed since the last frame, the above steps are mirrored using the previous beat period as the delay of comb filter 2. This is depicted by the right hand side branch in Fig. 6. The motivation for this is that if the prediction for the beat period in the current frame is erroneous, the comb filter tuned to the previous beat period may indicate this by remaining locked to the previous beat period and phase, and producing a more energetic output and thus larger score than the filter tuned to the erroneous current period.

In the final step, the best scores delivered by both branches are compared, and the one giving the largest score determines the final beat period and phase. Thus, if the comb filter branch tuned to the previous beat period gives a larger score, the beat period estimate is adjusted equal to the previous beat period. The state of the winning comb filter is stored to be used in the next frame as comb filter 2.

After the beat period and phase are obtained, the beat and tatum locations for the current audio frame are interpolated. Although this reduces the ability of the system to follow rapid tempo changes, it reduces the computational load since the back end processing is done only once for each audio frame.

3. Implementation

The authors have written a real-time implementation of the proposed method for the S60 smartphone platform. The implementation uses fixed-point arithmetic, where all signals are represented as 32-bit integers and coefficients as 16-bit integers. The power estimation low-pass filter is implemented simply as a first-order IIR due to the arithmetic used. Increasing the filter order would have a positive impact on performance, but the given filter design causes that the coefficients exceed 16-bit dynamic scale. Naturally, the accent power compression is realized by a 200-point lookup table. Tables are used also in the period and phase estimation for efficiently computing weight function values. The continuity function, the priors, and the likelihood surface shown in Fig. 5 are stored into lookup tables. Lookup tables are also utilized for storing precalculated feedback gain values for the comb filters. For efficiency, both the autocorrelation and discrete cosine transform processes are implemented on top of a fast Fourier transform (FFT).

For low-latency real-time implementation, the algorithm is split into two execution threads. Referring to Fig. 1, a high-priority "front-end" thread runs the *resampling* and *accent filter bank* stages, feeding their results into a memory buffer. The front-end runs synchronously with other audio signal processing. *Periodicity estimation* and following stages are run in a low-priority "back-end" thread, which is signaled when a new accent frame is available from *buffering* stage. The lower priority allows the back-end processing to take a longer time without interrupting the audio processing, unlinking audio frame length and accent frame length.

4. Evaluation

The proposed algorithm is evaluated in two aspects, beat tracking performance and computational complexity. The methods of Klapuri *et al.* [3] and Scheirer [4] are used as a comparison, using the original authors' implementations. [1]

4.1. Performance

The performance was evaluated by analyzing 192 songs in CD audio quality. Songs with a steady beat were selected from various genres. The majority of songs were rock/pop (43%), soul/R&B/funk (18%), jazz/blues (16%), and electronic/dance (11%) music, and all except two songs were in 4/4 meter. The beats of approximately one minute long song

[1] We wish to thank Anssi Klapuri and Eric Scheirer for making their algorithm implementations available for the comparison.

Table 1. Beat tracking accuracy scores.

Method	Continuity required			Individual estimates		
	Correct	Accept d/h	Period	Correct	Accept d/h	Period
Proposed	60%	70%	76%	64%	76%	79%
Klapuri	66%	76%	73%	72%	85%	81%
Scheirer	29%	34%	30%	53%	65%	59%

Table 2. Processor usage profiles.

Method	Mcycles	Mcycles/s
Proposed	678	2.3
Klapuri (MATLAB)	125000	420
Scheirer	136000	450
Scheirer without `malloc` etc.	119000	390

excerpts were annotated by tapping along with the song playing. The evaluation methodology followed the one proposed in [3], assessing both the period and phase estimation accuracy of the proposed method. A correct period estimate is defined to deviate less than 17.5% from the annotated reference, and the correct phase to deviate less than 0.175 times the annotated beat time. The following scores were calculated and averaged over the duration of the excerpts and over all 192 songs:

- Correct: Beat estimate with correct period and phase.
- Accept d/h: Beat estimate with period matching either 0.5, 1.0, or 2.0 times the correct value, and correct phase.
- Period: Beat estimate with correct period, phase is ignored.

We calculated the scores for both the longest continuous correctly analyzed segment and individual estimates without continuity requirement. For comparison, the methods proposed in [3] and [4] were run on the same data. The results are shown in Table 1. In summary, the proposed method approaches the Klapuri *et al.* method performance in all of the cases. The biggest deviations are in the Scheirer method scores with continuity requirement, reflecting the lack of beat period smoothing in the Scheirer method.

4.2. Complexity

We compared the computational complexity of the three algorithms on a PC having 1.86 GHz Pentium M processor and 1 GB of memory. The proposed and Scheirer methods were implemented in C++ in floating point and compiled with the same compiler settings; function inlining intrinsics were added into Scheirer's original algorithm. The Klapuri method is a combination of MATLAB and C++ code.

A 300-second audio clip was processed five times with each of the three methods and the algorithm CPU time was measured (excluding file access and decoding). The median CPU cycles of the five runs are shown in Table 2, divided by 10^6 (Mcycles), and normalized with audio clip length (Mcycles/s). The Klapuri method is not strictly comparable to the others because it is mostly MATLAB processing: 61% of the CPU is used in MATLAB code. The Scheirer method cycles break down into 82% for comb filtering and 13% for runtime functions (e.g. `malloc`). A second Scheirer profile in Table 2 has the runtime functions subtracted. The proposed algorithm is found over 170 times more efficient.

We also evaluated the computational complexity of the proposed method on a Nokia 6630 smartphone having a 220 MHz ARM9 processor. An instruction profiler was configured to sample the processor program counter on a 1 kHz rate, yielding 302500 data points in total. During playback, 13% of processor time was spent in the beat and tatum tracker implementation and 8% in MP3 format decoding. The profile shows the algorithm to perform very efficiently, comparable to the complexity of the MP3 decoder.

5. Conclusion

A beat and tatum tracker algorithm can be made computationally very efficient without compromising beat tracking performance. We introduced a novel beat and tatum tracker for music signals, consisting of multirate accent analysis, discrete cosine transform periodicity analysis, and phase estimation by adaptive comb filtering. The complexity of the proposed method is less than 1% of Scheirer's method, and its beat tracking accuracy approaches Klapuri's method. The authors have created a real-time implementation of the proposed method for the S60 smartphone platform.

References

[1] J.A. Bilmes. "Timing is of the Essence: Perceptual and Computational Techniques for Representing, Learning, and Reproducing Expressive Timing in Percussive Rhythm." M.Sc. Thesis, Massachusetts Institute of Tech., Sep. 1993.

[2] F. Gouyon and S. Dixon. "A review of automatic rhythm description systems." *Comp. Music J.*, 29(1):34–54, 2005.

[3] A.P. Klapuri, A.J. Eronen, and J.T. Astola. "Analysis of the meter of acoustic musical signals." *IEEE Trans. Audio, Speech, and Lang. Proc.*, 14(1):342–355, Jan. 2006.

[4] E.D. Scheirer. "Tempo and beat analysis of acoustic musical signals." *J. Acoust. Soc. Am.*, 103(1):588–601, Jan. 1998.

[5] J. Seppänen. "Tatum grid analysis of musical signals." In *Proc. IEEE Workshop on Applic. of Signal Proc. to Audio and Acoust. (WASPAA)*, pp. 131–134, New Paltz, NY, USA, Oct. 2001.

[6] C. Uhle, J. Rohden, M. Cremer, and J. Herre. "Low complexity musical meter estimation from polyphonic music." In *Proc. AES 25th Int. Conf.*, London, UK, 2004.

[7] P.P. Vaidyanathan. *Multirate Systems and Filter Banks.* Prentice-Hall, Upper Saddle River, NJ, USA, 1993.

Publication 7

A. Eronen, A. Klapuri, "Music Tempo Estimation using k-NN Regression", *IEEE Transactions on Audio, Speech, and Language Processing*, accepted for publication.

Music Tempo Estimation with k-NN Regression

*Antti Eronen and Anssi Klapuri

***Abstract*—An approach for tempo estimation from musical pieces with near-constant tempo is proposed. The method consists of three main steps: measuring the degree of musical accent as a function of time, periodicity analysis, and tempo estimation. Novel accent features based on the chroma representation are proposed. The periodicity of the accent signal is measured using the generalized autocorrelation function, followed by tempo estimation using k-Nearest Neighbor regression. We propose a resampling step applied to an unknown periodicity vector before finding the nearest neighbors. This step improves the performance of the method significantly. The tempo estimate is computed as a distance-weighted median of the nearest neighbor tempi. Experimental results show that the proposed method provides significantly better tempo estimation accuracies than three reference methods.**

***Index Terms*—Music tempo estimation, chroma features, k-Nearest Neighbor regression.**

I. Introduction

Musical meter is a hierarchical structure, which consists of pulse sensations at different time scales. The most prominent level is the *tactus*, often referred as the foot tapping rate or beat. The *tempo* of a piece is defined as the rate of the tactus pulse. It is typically represented in units of beats per minute (BPM), with a typical tempo being of the order of 100 BPM.

Human perception of musical meter involves inferring a regular pattern of pulses from moments of musical stress, a.k.a. *accents* [1, p.17]. Accents are caused by various events in the musical surface, including the beginnings of all discrete sound events, especially the onsets of long pitched sounds, sudden changes in loudness or timbre, and harmonic changes. Many automatic tempo estimators try to imitate this process to some extent: measuring musical accentuation, estimating the periods and phases of the underlying pulses, and choosing the level corresponding to the tempo or some other metrical level of interest [2].

Tempo estimation has many applications, such as making seamless "beatmixes" of consecutive music tracks with the help of beat alignment and time stretching. In disc jockey applications metrical information can be used to automatically locate suitable looping points. Visual appeal can be added to music players with beat synchronous visual effects such as virtual dancing characters. Other applications include finding music with certain tempo from digital music libraries in order to match the mood of the listener or to provide suitable motivation for the different phases of a sports exercise. In addition, automatically extracted beats can be used to enable musically-synchronized feature extraction for the purposes of structure analysis [3] or cover song identification [4], for example.

A. Previous work

Tempo estimation methods can be divided into two main categories according to the type of input they process. The earliest ones processed symbolic (MIDI) input or lists of onset times and durations, whereas others take acoustic signals as input. Examples of systems processing symbolic input include the ones by Rosenthal [5] and Dixon [6].

One approach to analyze acoustic signals is to perform discrete onset detection and then use e.g. inter onset interval (IOI) histogramming to find the most frequent periods, see e.g. [7], [8]. However, it has been found better to measure musical accentuation in a continuous manner instead of performing discrete onset detection [9]. A time-frequency representation such as energies at logarithmically distributed subbands is usually used to compute features that relate to the accents [2], [10]. This typically involves differentiation over time within the bands. Alonso *et al.* use a subspace analysis method to perform harmonic+noise decomposition before accent feature analysis [11]. Peeters proposes the use of a reassigned spectral energy flux [12], and Davies and Plumbley use the complex spectral difference [3].

Accent feature extraction is typically followed by periodicity analysis using e.g. the autocorrelation function (ACF) or a bank of comb-filter resonators. The actual tempo estimation is then done by picking one or more peaks from the periodicity vector, possibly weighted with the prior distribution of beat periods [2], [13], [10]. However, peak picking steps are error prone and one of the potential performance bottlenecks in rhythm analysis systems.

An interesting alternative to peak picking from periodicity vectors was proposed by Seyerlehner *et al.*, who used the k-Nearest Neighbor algorithm for tempo estimation [14]. Using the k-Nearest Neighbor algorithm was motivated based on the observation that songs with close tempi have similar periodicity functions. The authors searched the nearest neighbors of a periodicity vector and predicted the tempo according to the value that appeared most often within the k songs but did not report significant performance improvement over reference methods.

It should be noted that in the tempo estimation task, the temporal positions of the beats are irrelevant. In this sense, the present task differs from full meter analysis systems, where the positions of the beats need to be produced for example with dynamic programming [2], [10], [12], [15], [11] or Kalman filtering [16]. A full review of meter analysis systems is outside the scope of this article due to space restrictions. See [17] and [18] for more complete reviews.

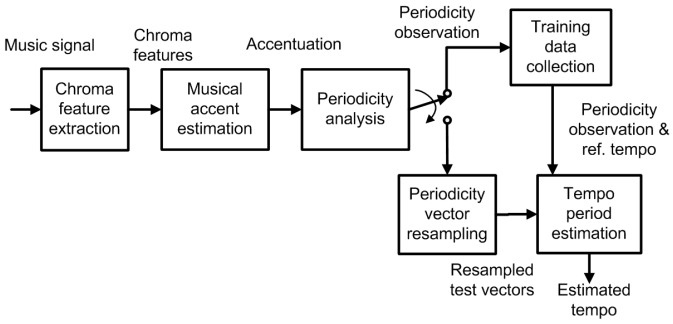


Fig. 1. Overview of the proposed method

B. Proposed method

In this paper, we study the use of the k-Nearest Neighbor algorithm for tempo estimation further. This is referred as k-NN regression as the tempo to be predicted is continuous-valued. Several improvements are proposed that significantly improve the tempo estimation accuracy using k-NN regression compared to the approach presented in [14]. First, if the training data does not have instances with very close tempi to the test instance, the tempo estimation is likely to fail. This is a quite common situation in tempo estimation because the periodicity vectors tend to be sharply peaked at the beat period and its multiples and because the tempo value to be predicted is continuous valued. With distance measures such as the Euclidean distance even small

A. Eronen is with Nokia Research Center, Finland, P.O. Box 100, FIN-33721 Tampere, Finland. E-mail: antti.eronen@nokia.com.

A. Klapuri is with the Department of Signal Processing, Tampere University of Technology, Finland. E-mail: anssi.klapuri@tut.fi.

Manuscript received Month XX, XXXX; revised Month XX, XXXX.

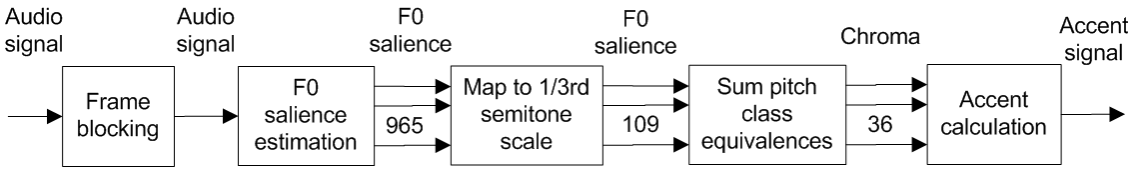


Fig. 2. Overview of musical accent analysis. The numbers between blocks indicate the data dimensionality if larger than one.

differences in the locations of the peaks in the periodicity vectors can lead to a large distance. We propose here a resampling step to be applied to the unknown test vector to create a set of test vectors with a range of possible tempi, increasing the likelihood of finding a good match from the training data. Second, to improve the quality of the training data we propose to apply an outlier removal step. Third, we observe that the use of locally weighted k-NN regression may further improve the performance.

The proposed k-NN regression based tempo estimation is tested using five different accent feature extractors to demonstrate the effectiveness of the approach and applicability across a range of features. Three of them are previously published and two are novel ones and use pitch chroma information. Periodicity is estimated using the generalized autocorrelation function which has previously been used for pitch estimation [19], [20]. The experimental results demonstrate that the chroma accent features perform better than three of the four reference accent features. The proposed method is compared to three reference methods and is shown to perform significantly better.

An overview of the proposed method is depicted in Figure 1. First, chroma features are extracted from the input audio signal. Then, accentuation is measured at different pitch classes, and averaged over the pitch classes to get a single vector representing the accentuation over time. Next, periodicity is analyzed from the accent signal. The obtained periodicity vector is then either stored as training data to be used in estimating tempo in the future (training phase), or subjected for resampling and tempo estimation (estimation phase). The following sections describe the various phases in detail.

II. Method

A. Musical accent analysis

1) Chroma feature extraction: The purpose of musical accent analysis is to extract features that effectively describe song onset information and discard information irrelevant for tempo estimation. In our earlier work [2], we proposed an accent feature extractor which utilizes 36 logarithmically distributed subbands for accent measurement and then folds the results down to four bands before periodicity analysis.

In this work, a novel accent analysis front end is described which further emphasizes the onsets of pitched events and harmonic changes in music and is based on the chroma representation used earlier for music structure analysis in [21]. Figure 2 depicts an overview of the proposed accent analysis. The chroma features are calculated using a multiple fundamental frequency (F0) estimator [22]. The input signal sampled at 44.1 kHz sampling rate and 16-bit resolution is first divided into 93 ms frames with 50% overlap. In each frame, the salience, or strength, of each F0 candidate is calculated as a weighted sum of the amplitudes of its harmonic partials in a spectrally whitened signal frame. The range of fundamental frequencies used here is 80 – 640 Hz. Next, a transform is made into a musical frequency scale having a resolution of 1/3rd-semitone (36 bins per octave). This transform is done by retaining only the maximum-salience fundamental frequency component for each 1/3rd of a semitone range. Finally the octave equivalence classes are summed over the whole pitch range using a resolution of three bins per semitone to produce a 36 dimensional chroma vector $x_b(k)$, where k is the frame index and $b = 1, 2, ..., b_0$ is the pitch class index, with $b_0 = 36$. The matrix $x_b(k)$ is normalized by removing the mean and normalizing the standard deviation of each chroma coefficient over time, leading to a normalized matrix $\widehat{x}_b(k)$.

2) Musical accent calculation: Next, musical accent is estimated based on the normalized chroma matrix $\widehat{x}_b(k)$, $k = 1, ..., K$, $b = 1, 2, ..., b_0$, much in a similar manner as proposed in [2], the main difference being that frequency bands are replaced with pitch classes. First, to improve the time resolution, the chroma coefficient envelopes are interpolated by a factor eight by adding zeros between the samples. This leads to the sampling rate $f_r = 172$ Hz. The interpolated envelopes are then smoothed by applying a sixth-order Butterworth low-pass filter (LPF) with $f_{LP} = 10$ Hz cutoff. The resulting smoothed signal is denoted by $z_b(n)$. This is followed by half wave rectification and weighted differentiation steps. A half-wave rectified (HWR) differential of $z_b(n)$ is first calculated as

$$z'_b(n) = \text{HWR}(z_b(n) - z_b(n-1)), \tag{1}$$

where the function $\text{HWR}(x) = \max(x, 0)$ sets negative values to zero and is essential to make the differentiation useful. Next we form a weighted average of $z_b(n)$ and its differential $z'_b(n)$:

$$u_b(n) = (1-\lambda)z_b(n) + \lambda \frac{f_r}{f_{\text{LP}}} z'_b(n), \tag{2}$$

where $0 \leq \lambda \leq 1$ determines the balance between $z_b(n)$ and $z'_b(n)$, and the factor f_r/f_{LP} compensates for the small amplitude of the differential of a low-pass-filtered signal [2].

Finally, bands are linearly averaged to get a single accent signal $a(n)$ to be used for periodicity estimation. It represents the degree of musical accent as a function of time.

B. Periodicity analysis

Periodicity analysis is carried out on the accent signal. Several periodicity estimators have been proposed in the literature, such as the inter-onset interval histogramming [7], autocorrelation function (ACF) [23], or comb filter banks [24]. In this paper, we use the generalized autocorrelation function (GACF) which is computationally efficient and has proven to be a robust technique in multipitch analysis [20]. The GACF is calculated without windowing in successive frames of length W and 16% overlap. The input vector $\mathbf{a}_m$ at the mth frame has the length of $2W$ after zero padding to twice its length:

$$\mathbf{a}_m = [a((m-1)W), ..., a(mW-1), 0, ..., 0]^{\text{T}}, \tag{3}$$

where T denotes transpose. The GACF is defined as ([19]):

$$\rho_m(\tau) = \text{IDFT}(|\text{DFT}(\mathbf{a}_m)|^p), \tag{4}$$

where DFT stands for Discrete Fourier Transform and IDFT its inverse. The coefficient p controls the frequency domain compression. $\rho_m(\tau)$ gives the strength of periodicity at period (lag) τ. The GACF

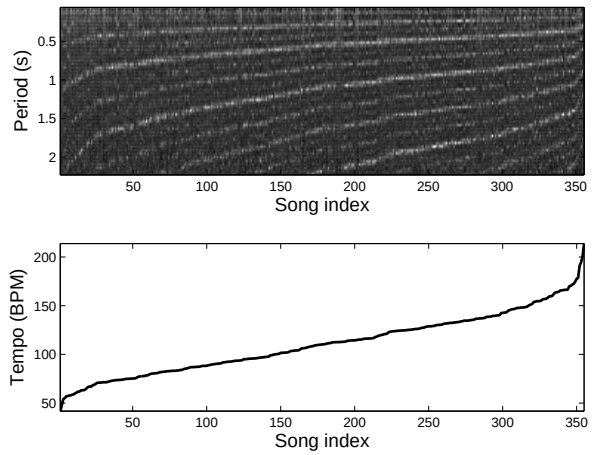


Fig. 3. Upper panel: periodicity vectors of musical excerpts in our evaluation dataset ordered in ascending tempo order. The shape of the periodicity vectors is similar across pieces, with the position of the peaks changing with tempo. Lower panel: corresponding annotated tempi of the pieces.

was selected because it is straightforward to implement as usually the fast Fourier transform routines are available, and it suffices to optimize the single parameter p to make the transform optimal for different accent features. The conventional ACF is obtained with $p = 2$. We optimized the value of p for different accent features by testing a range of different values and performing the tempo estimation on a subset of the data. The value that led to the best performance was selected for each feature. For the proposed chroma accent features, the value used was $p = 0.65$.

At this step we have a sequence of periodicity vectors computed in adjacent frames. If the goal is to perform beat tracking where the tempo can vary in time, we would consider each periodicity vector separately and estimate the tempo as a function of time from each vector separately. In this paper, we are interested in getting a single representative tempo value for each musical excerpt. Therefore, we obtain a single representative periodicity vector $\rho_{med}(\tau)$ for each musical excerpt by calculating point-wise median of the periodicity vectors over time. This assumes that the excerpt has nearly constant tempo and is sufficient in applications where a single representative tempo value is desired. The median periodicity vector is further normalized to remove the trend due to the shrinking window for larger lags

$$\hat{\rho}_{med}(\tau) = \frac{1}{W - \tau}\rho_{med}(\tau). \tag{5}$$

The final periodicity vector is obtained by selecting the range of bins corresponding to periods from 0.06 s to 2.2 s, and removing the mean and normalizing the standard deviation to unity for each periodicity vector.

The resulting vector is denoted by $s(\tau)$. Figure 3 presents the periodicity vectors for the songs in our evaluation database, ordered in ascending tempo order. Indeed, the shape of the periodicity vectors is similar across music pieces, with the position of the peaks changing with tempo.

C. Tempo estimation by k-NN regression

The tempo estimation is formulated here as a regression problem: given the periodicity observation $s(\tau)$, we estimate the continuous valued tempo T. In this paper, we propose to use locally weighted learning ([25]) to solve the problem. More specifically, we use k-Nearest Neighbors regression and compute the tempo as a weighted median of the nearest neighbor tempi. In conventional k-NN regression, the property value of an object is assigned to be the average of the values of its k nearest neighbors. The distance to the nearest neighbors is typically calculated using the Euclidean distance.

In this paper, several problem-specific modifications are proposed to improve the performance of tempo estimation using k-NN regression. First, a resampling step is proposed to alleviate problems caused by mismatches of the exact tempo values in the testing and training data. Distance measures such as the Euclidean distance or correlation distance are sensitive to whether the peaks in the unknown periodicity vector and the training vectors match exactly. With the resampling step it is more likely that similarly shaped periodicity vector(s) with a close tempi are found from the training set. Resampling is applied to "stretch" and "shrink" the unknown test vectors to increase the likelihood of a matching training vector to be found from the training set. Since the tempo values are continuous, the resampling ensures that we do not need to have a training instance with exactly the same tempo as the test instance in order to find a good match.

Thus, given a periodicity vector $s(\tau)$ with unknown tempo T, we generate a set of resampled test vectors $s_r(\tau)$, where subscript r indicates the resampling ratio. A resampled test vector will correspond to a tempo of T/r. We tested various possible ranges for the resampling ratio, and 15 linearly spaced ratios between 0.87 and 1.15 were taken into use. Thus, for a piece having a tempo of 120 BPM the resampled vectors correspond to a range of tempi from 104 to 138 BPM.

When receiving an unknown periodicity vector, we first create the resampled test vectors $s_r(\tau)$. The Euclidean distance between each training vector $t_m(\tau)$ and the resampled test vectors is calculated as

$$d(m, r) = \sqrt{\sum_{\tau} (t_m(\tau) - s_r(\tau))^2} \tag{6}$$

where $m = 1, ..., M$ is the index of the training vector. The minimum distance $d(m) = \min_r d(m, r)$ is stored for each training instance m, along with the resampling ratio that leads to the minimum distance $\widehat{r}(m) = \text{argmin}_r\, d(m, r)$. The k nearest neighbors that lead to the k lowest values of $d(m)$ are then used to estimate the unknown tempo. The annotated tempo $T_{\text{ann}}(i)$ of the nearest neighbor i is now an estimate of the resampled test vector tempo. Multiplying the nearest neighbor tempo with the ratio gives us an estimate of the original test vector tempo: $\widehat{T}(i) = T_{\text{ann}}(i)\widehat{r}(i)$.

The final tempo estimate is obtained as a weighted median of the nearest neighbor tempo estimates $\widehat{T}(i)$, $i = 1, ..., k$. Due to the weighting, training instances close to the test point have a larger effect on the final tempo estimate. The weights w_i for the k nearest neighbors are calculated as

$$w_i = \frac{\exp\left(-\gamma d(i)\right)}{\sum_{i=1}^{k} \exp\left(-\gamma d(i)\right)}, \tag{7}$$

where the parameter γ controls how steeply the weighting decreases with increasing distance d, and $i = 1, ..., k$. The value $\gamma = 40$ was found by monitoring the performance of the system with a subset of the data. The exponential function fulfils the requirements for a weighting function in locally weighted learning: the maximum value is at zero distance, and the function decays smoothly as the distance increases [25]. The tempo estimate is then calculated as a weighted median of the tempo estimates $\widehat{T}(i)$ using the weights w_i with the procedure described in [26]. The weighted median gives significantly better results than a weighted mean. The difference between weighted median and unweighted median is small but consistent in favor of the weighted median when the parameter γ is properly set.

In addition, the use of an outlier removal step is evaluated to

improve the quality of the training data. We implemented leave-one-out outlier removal as described in [27]. It works within the training data by removing each sample in turn from the training data, and classifying it by all the rest. Those training samples that are misclassified are removed from the training data.

III. Results

This section looks at the performance of the proposed method in simulations and compares the results to three reference systems and three accent feature extractors.

A. Experimental setup

A database of 355 musical pieces with CD quality audio was used to evaluate the system and the three reference methods. The musical pieces were a subset[1] of the material used in [2]. The database contains examples of various musical genres whose distribution is the following: 82 classical pieces, 28 electronic/dance, 12 hip hop/rap, 60 jazz/blues, 118 rock/pop, 42 soul/RnB/funk, and 13 world/folk. Full listing of the database is available at www.cs.tut.fi/~eronen/taslp08-tempo-dataset.html. The beat was annotated from approximately one-minute long representative excerpts by a musician who tapped along with the pieces. The ground truth tempo for each excerpt is calculated based on the median inter-beat-interval of the tapped beats. The distribution of tempi is depicted in figure 4.

We follow here the evaluation presented in [14]. Evaluation is done using leave-one-out cross validation: the tempo of the unknown song is estimated using all the other songs in the database. The tempo estimate is defined to be correct if the predicted tempo estimate is within 4% of the annotated tempo.

Along with the tempo estimation accuracy, we also report a tempo category classification accuracy. Three tempo categories were defined: from 0 to 90 BPM, 90 to 130 BPM, and above 130 BPM. Classification of the tempo category is considered successful if the predicted tempo falls within the same category as the annotated tempo. This kind of "rough" tempo estimate is useful in applications that would only require e.g. classifying songs to slow, medium, and fast categories.

The decision whether the differences in error rates is statistically significant is done using McNemar's test [28]. The test assumes that the trials are independent, an assumption that holds in our case since the tempo estimation trials are performed on different music tracks. The null hypothesis H_0 is as follows: given that only one of the two algorithms makes an error, it is equally likely to be either one. Thus, this test considers those trials where two systems make different predictions, since no information on their relative difference is available from trials in which they report the same outcome. The test is calculated as described in [28, Section 3], and H_0 is rejected if the P-value is less than a selected significance level α. We report the results using the following significance levels and wordings: $P \geq 0.05$, not significant (NS); $0.01 \leq P < 0.05$, significant (S); $0.0001 \leq P < 0.01$, very significant (VS); and $P < 0.0001$, highly significant (HS).

B. Reference methods

To put the results in perspective, the results are presented in comparison to three reference methods. The first was described by Ellis [10], and is based on an accent feature extractor using the mel-frequency filterbank, autocorrelation periodicity estimation, and dynamic programming to find the beat times. The implementation is also provided by Ellis [29]. The second reference method was proposed by ourselves in [2] and was the best performing method in the Music Information Retrieval Evaluation eXchange (MIREX 2006) evaluations [9]. The third has been described in [13] and is based on a computationally efficient accent feature extraction based on multirate analysis, discrete cosine transform periodicity analysis, and period determination utilizing simplified musicological weight functions. The comparison against the Ellis method may not be completely fair as it has not received any parameter optimization on any subset of the data used. However, the two other methods have been developed on the same data and are thus good references.

[1]The subset consisted of all music tracks to which the first author had access.

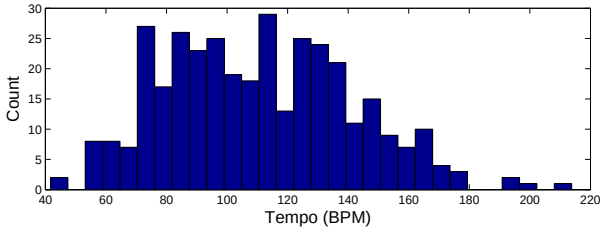


Fig. 4. Distribution of the annotated tempi in the evaluation database.

In addition to comparing the performance of the proposed method to the complete reference systems, we also evaluate the proposed musical accent measurement method against four other features. This is done by using the proposed k-NN regression tempo estimation with accent features proposed elsewhere. Comparisons are presented to two auditory spectrogram based accent features: first using a critical band scale as presented in [2] (KLAP) and the second using the Mel-frequency scale (MEL). Another two accent features are based on the quadrature mirror filter bank of [13] (QMF), and a straightforward chroma feature analysis (SIMPLE). The main difference between the various methods is how the frequency decomposition is done, and how many accent bands are used for periodicity analysis. In the case of the MEL features, the chroma vector $x_b[k]$ is replaced with the output band powers of the corresponding auditory filterbank. In addition, logarithmic compression is applied to the band envelopes before the interpolation step, and each nine adjacent accent bands are combined into one resulting into four accent bands. Periodicity analysis is done separately for four bands, and final periodicity vector is obtained by summing across bands. See the details in [2]. In the case of the QMF and KLAP front ends, the accent feature calculation is as described in the original publications [13] and [2]. The method SIMPLE differs from the method proposed in this paper in how the chroma features are obtained: whereas the proposed method uses saliences of F0 estimates mapped on a musical scale, the method SIMPLE simply accumulates the energy of FFT bins to 12 semitone bins. The accent feature parameters such as λ were optimized for both the chroma accent features and the MEL accent features using a subset of the data. The parameters for the KLAP and QMF methods are as presented in the original publications [13] and [2]. The frame size and frame hop for the methods MEL and SIMPLE is fixed at 92.9 ms and 46.4 ms, respectively. The KLAP feature extractor utilizes a frame size of 23 ms with 50% overlap.

C. Experimental results

1) Comparison to reference methods: Table I shows the results of the proposed method in comparison with the reference systems. The statistical significance is reported under each accuracy percentage in comparison to the proposed method. All the reference systems output both the period and timing of the beat time instants and the output tempo is calculated based on the median inter beat interval. We

TABLE I
RESULTS IN COMPARISON TO REFERENCE METHODS. THE STATISTICAL TESTS ARE DONE IN COMPARISON TO THE PROPOSED METHOD IN THE LEFTMOST COLUMN.

	Proposed	Ellis [10]	Seppänen *et al.* [13]	Klapuri *et al.* [2]
Tempo	79%	45%	64%	71%
Significance	-	HS	HS	HS
Tempo category	77%	52%	64%	68%
Significance	-	HS	HS	VS

TABLE II
RESULTS WITH DIFFERENT ACCENT FEATURE EXTRACTORS.

	Proposed	KLAP	SIMPLE	MEL	QMF
Tempo	79%	76%	73%	75%	63%
Significance	-	NS	S	HS	HS
Tempo category	77%	75%	75%	74%	72%
Significance	-	NS	NS	VS	S

TABLE III
RESULTS WHEN DISABLING CERTAIN STEPS. COMPARE THE RESULTS TO THE COLUMN "PROPOSED" OF TABLES I AND II.

	No resamp.	No outlier rem.	Plain median
Tempo	75%	78%	77%
Significance	S	NS	NS
Tempo category	72%	79%	76%
Significance	VS	NS	NS

observe a highly significant or very significant performance difference in comparison to all the reference methods in both tasks.

2) Importance of different elements of the proposed method: The following experiments study the importance of different elements of the proposed method in detail. Table II presents the results obtained using different accent feature extractors. The performance of a certain accent feature extractor depends on the parameters used, such as the parameter λ controlling the weighted differentiation described in section II-A2. There is also some level of dependency between the accent features and periodicity estimation parameters, i.e. the length of the GACF window, and the exponent used in computing the GACF. These parameters were optimized for all accent features using a subset of the database, and the results are reported for the best parameter setting.

The proposed chroma accent features based on F0 salience estimation perform best, although the difference is not statistically significant in comparison to the accent features proposed earlier in [2]. The difference in comparison to the three other front ends in tempo estimation is statistically significant. The accent features based on the QMF-decomposition are computationally very attractive and may be a good choice if the application only requires classification into rough tempo categories, or if the music consists mainly of material with a strong beat.

Table III shows the results when the resampling step in tempo regression estimation or the outlier removal step is disabled, or when no weighting is used when computing the median of nearest neighbor tempo estimates. The difference in performance when the resampling step is removed is significant. Our explanation for this is that without the resampling step it is quite unlikely that similarly shaped example(s) with close tempi are found from the training set, and even small differences in the locations of the peaks in the periodicity vector can lead to a large distance.

TABLE IV
CONFUSION MATRIX IN CLASSIFYING INTO TEMPO CATEGORIES SLOW (0 TO 90 BPM), MEDIUM (90 TO 130 BPM), AND FAST (OVER 130 BPM) FOR THE PROPOSED METHOD. ROWS CORRESPOND TO ANNOTATED TEMPO CATEGORIES, COLUMNS TO ESTIMATED TEMPO CATEGORIES.

	slow	medium	fast
slow	76%	16%	8%
medium	4%	96%	0%
fast	28%	14%	58%

TABLE V
CONFUSION MATRIX IN CLASSIFYING INTO TEMPO CATEGORIES FOR THE REFERENCE METHOD KLAPURI *et al.* [2]. ROWS CORRESPOND TO ANNOTATED TEMPO CATEGORIES, COLUMNS TO ESTIMATED TEMPO CATEGORIES.

	slow	medium	fast
slow	60%	30%	10%
medium	1%	99%	0%
fast	32%	24%	44%

The outlier removal step does not have statistically significant effect on the performance when using the chroma features. However, this is the case only with the chroma features for which the result is shown here. The accuracy obtained using the chroma features is already quite good and the outlier removal step is not able to improve from that. For all other features the outlier removal improves the performance in both tempo and tempo category classification by several percentage points (the results in Table II are calculated with outlier removal enabled). Using distance based weighting in the median calculation gives a small but not statistically significant improvement in the accuracy.

3) Performance across tempo categories: Examining the performance across in classifying within different tempo categories is illustrative of the performance of the method, showing how evenly the method performs with slow, medium, and fast tempi. Tables IV and V depict the confusion matrices in tempo category classification for the proposed method and the best performing reference method, respectively. Rows correspond to presented tempo, columns to the estimated tempo category. Errors with slow and fast tempi cause the accuracy of tempo category classification to be generally smaller than that of tempo estimation. Both methods perform very well in classifying the tempo category within the medium range of 90 to 130 BPM. However, especially fast tempi are often underestimated by a factor of two: the proposed method would still classify 28% of fast pieces as slow. Very fast tempi might deserve special treatment in future work.

4) Effect of training data size: The quality and size of the training data has an effect on the performance of the method. To test the effect of the training data size, we ran the proposed method while varying the size of the training data. The outlier removal step is omitted. Figure 5 shows the result of this experiment. Uniform random samples with a fraction of the size of the complete training data were used to perform classification. A graceful degradation in performance is observed. The drop in performance becomes statistically significant at training data size of 248 vectors, however, over 70% accuracy is obtained using only 71 reference periodicity vectors. Thus, good performance can be obtained with small training data sizes if the reference vectors span the range of possible tempi in a uniform manner.

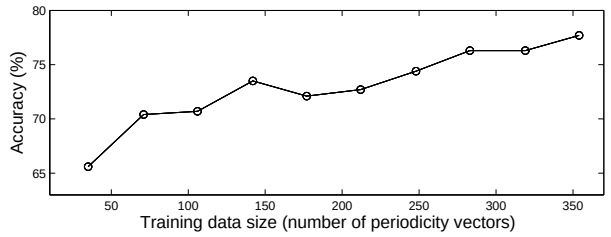


Fig. 5. Effect of training data size (number of reference periodicity vectors) on tempo estimation accuracy.

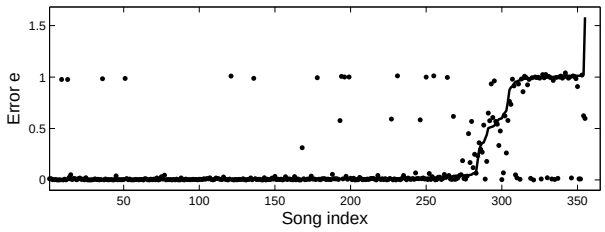


Fig. 6. Comparison of errors made by the proposed method using the chroma accent features (solid line) and the KLAP accent features (dot). The excerpts are ordered according to increasing error made by the proposed method, thus the order is different than in figure 3.

5) Using an artist filter: There are some artists in our database which have more than one music piece. We made a test using the so-called artist filter to ensure that this does not have a positive effect on the results. Pampalk has reported that using an artist filter is essential for not to overtrain a musical genre classifier [30]. We reran the simulations of the proposed method and, in addition to the test song, excluded all songs from the same artist. This did not have any effect on the correctly estimated pieces. Thus, musical pieces from the same artist do not overtrain the system.

6) Computational complexity: To get a rough idea of the computational complexity of the method, a set of 50 musical excerpts were processed with each of the methods and the total run time was measured. From fastest to slowest, the total run times are 130 seconds for Seppänen *et al.* [13], 144 seconds for the proposed method, 187 seconds for Ellis [10], and 271 seconds for Klapuri *et al.* [2]. The Klapuri *et al.* method was the only one that was implemented completely in C++. The Seppänen *et al.* and Ellis methods were Matlab implementations. The accent feature extraction of the proposed method was implemented in C++, the rest in Matlab.

IV. Discussion and future work

Several potential topics exist for future research. There is some potential for further improving the accuracy by combining different types of features as suggested by one of the reviewers. Figure 6 presents a pairwise comparison of the two best performing accent front ends: the F0-salience based chroma accent proposed in this paper and the method KLAP. The songs have been ordered with respect to increasing error made by the proposed method. The error is computed as follows ([9]):

$$e = |\log2(\frac{\text{computed tempo}}{\text{correct tempo}})|. \quad (8)$$

The value 0 corresponds to correct tempo estimates, and the value 1 to tempo halving or doubling. Out of the 355 test instances, 255 instances were correctly estimated using both accent features. 60 instances were incorrectly estimated using both accent features. At indices between 310 and 350 the method KLAP correctly estimates some cases where the proposed method makes tempo doubling or halving errors. But at the same range there are also many cases where the estimate is wrong using both accent features. Nevertheless, there is some complementary information in these accent feature extractors which might be utilized in the future.

Second direction is to study whether a regression approach can be implemented for beat phase and barline estimation. In this case, a feature vector is constructed by taking values of the accent signal during a measure, and the beat or measure phase is then predicted using regression with the collected feature vectors. Chroma is generally believed to highlight information on harmonic changes ([31]), thus the proposed chroma accent features would be worth testing in barline estimation.

V. Conclusion

A robust method for music tempo estimation was presented. The method estimates the tempo using locally weighted k-NN regression and periodicity vector resampling. Good performance was obtained by combining the proposed estimator with different accent feature extractors.

The proposed regression approach was found to be clearly superior compared to peak picking techniques applied on the periodicity vectors. We conclude that most of the improvement is attributed to the regression based tempo estimator with a smaller contribution to the proposed F0-salience chroma accent features and GACF periodicity estimation, as there is no statistically significant difference in error rate when the accent features used in [2] are combined with the proposed tempo estimator.

In addition, the proposed regression approach is straightforward to implement and requires no explicit prior distribution for the tempo as the prior is implicitly included in the distribution of the k-NN training data vectors. The accuracy degrades gracefully when the size of the training data is reduced.

References

[1] F. Lerdahl and R. Jackendoff, *A Generative Theory of Tonal Music*. Cambridge, MA, USA: MIT Press, 1983.

[2] A. P. Klapuri, A. J. Eronen, and J. T. Astola, "Analysis of the meter of acoustic musical signals," *IEEE Trans. Speech and Audio Proc.*, vol. 14, no. 1, pp. 342–355, Jan. 2006.

[3] M. E. Davies and M. D. Plumbley, "Context-dependent beat tracking of musical audio," *IEEE Trans. Audio, Speech, and Language Proc.*, pp. 1009–1020, Mar. 2007.

[4] J. Jensen, M. Christensen, D. Ellis, and S. Jensen, "A tempo-insensitive distance measure for cover song identification based on chroma features," in *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, Mar. 2008, pp. 2209–2212.

[5] D. F. Rosenthal, "Machine rhythm: Computer emulation of human rhythm perception," Ph.D. Thesis, Massachusetts Institute of Tech., Aug. 1992.

[6] S. Dixon, "Automatic extraction of tempo and beat from expressive performances," *J. New Music Research*, vol. 30, no. 1, pp. 39–58, 2001.

[7] J. Seppänen, "Tatum grid analysis of musical signals," in *Proc. IEEE Workshop on Applicat. of Signal Proc. to Audio and Acoust. (WASPAA)*, New Paltz, NY, USA, Oct. 2001, pp. 131–134.

[8] F. Gouyon, P. Herrera, and P. Cano, "Pulse-dependent analyses of percussive music," in *Proc. AES 22nd Int. Conf.*, Espoo, Finland, 2002.

[9] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano, "An experimental comparison of audio tempo induction algorithms," *IEEE Trans. Audio, Speech, and Language Proc.*, vol. 14, no. 5, pp. 1832–1844, 2006.

[10] D. P. Ellis, "Beat tracking by dynamic programming," *J. New Music Research*, vol. 36, no. 1, pp. 51–60, 2007.

[11] M. Alonso, G. Richard, and B. David, "Accurate tempo estimation based on harmonic+noise decomposition," *EURASIP J. Adv. in Signal Proc.*, 2007.

[12] G. Peeters, "Template-based estimation of time-varying tempo," *EURASIP J. Adv. in Signal Proc.*, no. 1, pp. 158–171, Jan. 2007.

[13] J. Seppänen, A. Eronen, and J. Hiipakka, "Joint beat & tatum tracking from music signals," in *7th International Conference on Music Information Retrieval (ISMIR-06)*, Victoria, Canada, 2006.

[14] K. Seyerlehner, G. Widmer, and D. Schnitzer, "From rhythm patterns to perceived tempo," in *8th International Conference on Music Information Retrieval (ISMIR-07)*, Vienna, Austria, 2007.

[15] D. Eck, "Beat tracking using an autocorrelation phase matrix," in *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, 2007, pp. 1313–1316.

[16] Y. Shiu and C.-C. J. Kuo, "Musical beat tracking via kalman filtering and noisy measurements selection," in *Proc. IEEE Int. Symp. Circ. and Syst.*, May 2008, pp. 3250–3253.

[17] F. Gouyon and S. Dixon, "A review of automatic rhythm description systems," *Comp. Music J.*, vol. 29, no. 1, pp. 34–54, 2005.

[18] S. Hainsworth, "Beat tracking and musical metre analysis," in *Signal Processing Methods for Music Transcription*, A. Klapuri and M. Davy, Eds. New York, NY, USA: Springer, 2006, pp. 101–129.

[19] H. Indefrey, W. Hess, and G. Seeser, "Design and evaluation of double-transform pitch determination algorithms with nonlinear distortion in the frequency domain-preliminary results," in *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, vol. 10, Apr. 1985, pp. 415–418.

[20] T. Tolonen and M. Karjalainen, "A computationally efficient multipitch analysis model," *IEEE Trans. Speech and Audio Proc.*, vol. 8, no. 6, pp. 708–716, 2000.

[21] J. Paulus and A. Klapuri, "Music structure analysis using a probabilistic fitness measure and an integrated musicological model," in *Proc. of the 9th International Conference on Music Information Retrieval (ISMIR 2008)*, Philadelphia, Pennsylvania, USA, 2008.

[22] A. Klapuri, "Multiple fundamental frequency estimation by summing harmonic amplitudes," in *7th International Conference on Music Information Retrieval (ISMIR-06)*, Victoria, Canada, 2006.

[23] C. Uhle, J. Rohden, M. Cremer, and J. Herre, "Low complexity musical meter estimation from polyphonic music," in *Proc. AES 25th Int. Conf.*, London, UK, 2004.

[24] E. D. Scheirer, "Tempo and beat analysis of acoustic musical signals," *J. Acoust. Soc. Am.*, vol. 103, no. 1, pp. 588–601, Jan. 1998.

[25] C. Atkeson, A. Moore, and S. Schaal, "Locally weighted learning," *AI Review*, vol. 11, pp. 11–73, Apr. 1997.

[26] Y. Lin, Y. Ruikang, M. Gabbouj, and Y. Neuvo, "Weighted median filters: a tutorial," *IEEE Trans. on Circuits and Systems II: Analog and Digital Signal Proc.*, vol. 43, no. 3, pp. 157–192, 1996.

[27] A. A. Livshin, G. Peeters, and X. Rodet, "Studies and improvements in automatic classification of musical sound samples," in *Proc. Int. Computer Music Conference (ICMC 2003)*, Singapore, 2003.

[28] L. Gillick and S. Coz, "Some statistical issues in the comparison of speech recognition algorithms," in *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP)*, vol. 1, 1989, pp. 532–535.

[29] D. P. Ellis, "Music beat tracking software." [Online]. Available: http://labrosa.ee.columbia.edu/projects/coversongs/

[30] E. Pampalk, "Computational models of music similarity and their application in music information retrieval," Ph.D. dissertation, Vienna University of Technology, Vienna, Austria, March 2006. [Online]. Available: http://www.ofai.at/~elias.pampalk/publications/pampalk06thesis.pdf

[31] M. Goto, "Real-time beat tracking for drumless audio signals: Chord change detection for musical decisions," *Speech Communication*, vol. 27, no. 3–4, pp. 311–335.

Publication 8

A. Eronen, "Chorus detection with combined use of MFCC and chroma features and image processing filters", *Proceedings of the 10th International Conference on Digital Audio Effects, DAFx-07*, Bordeaux, France, September 2007.

CHORUS DETECTION WITH COMBINED USE OF MFCC AND CHROMA FEATURES AND IMAGE PROCESSING FILTERS

Antti Eronen
Nokia Research Center
Tampere, Finland
Antti.Eronen@nokia.com

ABSTRACT

A computationally efficient method for detecting a chorus section in popular and rock music is presented. The method utilizes a distance matrix representation that is obtained by summing two separate distance matrices calculated using the mel-frequency cepstral coefficient and pitch chroma features. The benefit of computing two separate distance matrices is that different enhancement operations can be applied on each. An enhancement operation is found beneficial only for the chroma distance matrix. This is followed by detection of the off-diagonal segments of small distance from the distance matrix. From the detected segments, an initial chorus section is selected using a scoring mechanism utilizing several heuristics, and subjected to further processing. This further processing involves using image processing filters in a neighborhood of the distance matrix surrounding the initial chorus section. The final position and length of the chorus is selected based on the filtering results. On a database of 206 popular & rock music pieces an average F-measure of 86% is obtained. It takes about ten seconds to process a song with an average duration of three to four minutes on a Windows XP computer with a 2.8 GHz Intel Xeon processor.

1. INTRODUCTION

Music thumbnailing refers to the extraction of a characteristic, representative excerpt from a music file. Often the chorus or refrain is the most representative and "catchiest" part of a song. A basic application is to use this excerpt for previewing a music track. This is very useful if the user wishes to quickly get an impression of the content of a playlist, for example, or quickly browse the songs in an unknown album. In addition, the chorus part of a song would often make a good ring tone for a mobile phone, and automatic analysis of the chorus section would thus facilitate extraction of ring tone sections from music files.

Western popular music is well suited for automatic thumbnailing as it often consists of distinguishable sections, such as intro, verse, chorus, bridge, and outro. For example, the structure of a song may be intro, verse, chorus, verse, chorus, chorus. Some songs do not have as clear verse-chorus structure but there still are separate sections, such as section A and section B that repeat. In this case the most often repeating and energetic section is likely to contain the most recognizable part of the song.

Peeters et al. ([1]) divide the methods for chorus detection and music structure analysis into two main categories: the "state approach" which is based on clustering feature vectors to states having distinctive statistics, and the "sequence approach" which is based on computing a self-similarity matrix for the signal. One of the first examples of the state approach was that of Logan and Chu [2]. Recently, e.g. Levy et al. [3] and Rhodes et al. [4] have studied this approach. Similarity-matrix based approaches include the ones by Wellhausen & Crysandt [5] and Cooper & Foote [6]. Bartsch & Wakefield [7] and Goto [8] operated on an equivalent time-lag triangle representation. There are also methods utilizing many different cues, including e.g. segmentation into vocal / nonvocal sections, such as [9], or methods that iteratively try to find an optimal segmentation [10].

Here we present a method for detecting the chorus or some other often repeating and representative section from music files. The method is based on the self-similarity (distance) representation. The goal was to device a computationally efficient method that still would produce high quality music thumbnails for practical applications. Thus, iterative methods based on feature clustering or computationally intensive optimization steps could not be used. The following summarizes the novel aspects of the proposed method:

The self-distance matrix (SDM) used in the system is obtained by summing two distance matrices calculated using MFCC and chroma features. This improves the performance compared to the case when either of the features would be used alone. Although the MFCC features are sensitive to changing instrumentation between the occurrences of the chorus, the fact that the instrumentation and expression during the chorus is often different than in other parts of the song seems to overweigh this, at least with our pop & rock dominated data. The benefit of the proposed distance-matrix summing approach instead of merely concatenating the features into one, longer vector is that different enhancement operations can be applied for each matrix.

An initial chorus section is obtained from the repetitions detected from the SDM by utilizing a novel heuristic scoring scheme. The heuristics consider aspects such as the position of a repetition in the self-distance matrix (SDM), the adjustment of a repetition in relation to other repetitions in the SDM, average energy and average distance in the SDM during the repetition, and number of times the repetition occurs in the musical data.

The system performs the chorus determination in two steps: first a preliminary candidate is found for the chorus section, and then its final location and duration is determined by filtering with a set of image processing filters, selecting the final chorus position and duration according to the filter which gives the best fit.

Evaluations are presented on a database of 206 popular and rock music pieces. The method is demonstrated to provide sufficient accuracy for practical applications while being computationally efficient.

2. METHOD

2.1. Overview

Figure 1 shows an overview of the proposed method, which consists of the following steps. First the beats of the music signal are detected. Then, beat synchronous mel-frequency cepstral coefficient (MFCC) and pitch chroma features are calculated. This results in a sequence of MFCC and chroma feature vectors. Next, two self-distance matrices (SDM) are calculated, one for the MFCC features and one for the chroma features. Each item in the SDM represents the distance of feature vector at beat *i* to a feature vector at beat *j*. In the distance matrix representation, choruses or other repeating sections are shown as diagonal lines of small distance. The diagonal lines of the chroma distance matrix are then enhanced. Next we obtain a summed distance matrix by summing the chroma and MFCC distance matrices. This is followed by binarization of the summed distance matrix, which attempts to detect the diagonal regions of small distance (or high similarity). From the detected diagonal segments, the most likely chorus section candidate (diagonal stripe) is selected, and subjected to further processing. This further processing involves using image processing filters in a neighbourhood of the similarity matrix which surrounds the most likely chorus candidate. The final position and length of the chorus is selected based on the image processing results.

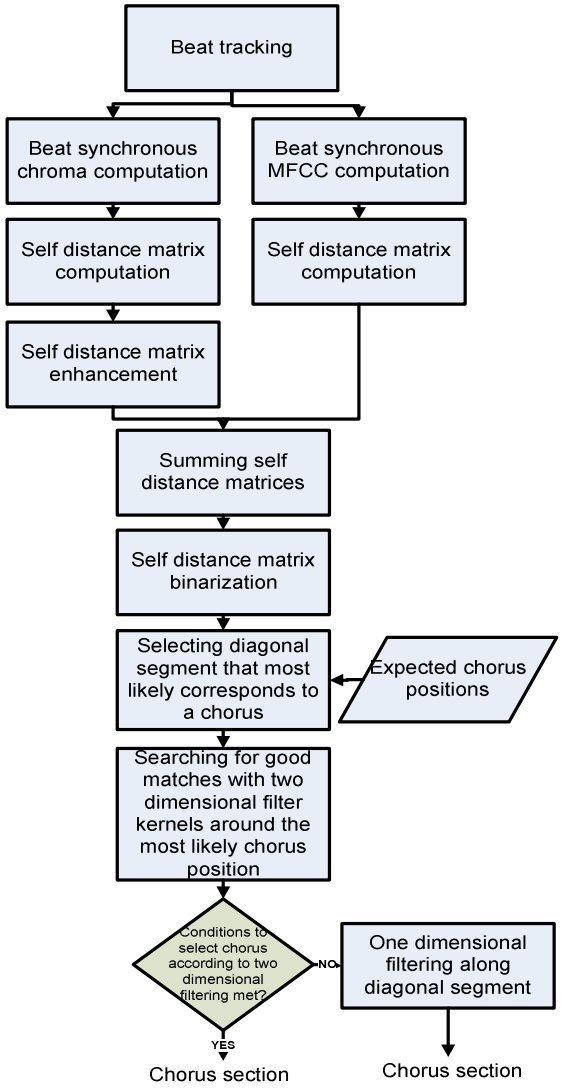


Figure 1: *Overview of the proposed method.*

2.2. Beat tracking

The feature extraction step begins by finding the beats in the acoustic music signal. We utilize the efficient beat tracking method described in [11] to produce an initial set of beat times and an accent signal $v(n)$. The accent signal measures the change in the spectrum of the signal and exhibits peaks at onset locations. An additional, non-causal postprocessing step was implemented to prevent the beat interval from changing significantly from one frame to another, which might cause problems with the beat synchronous self-distance matrices. The postprocessing is performed with a dynamic programming method described by Ellis [12]. The dynamic programming step takes as input the accent signal and median beat period produced by the method described in [11], performs smoothing of the accent signal with a Gaussian window, and then finds the optimal sequence of beats through the smoothed accent signal. The method iterates through each sample of the smoothed accent signal, and finds the best previous beat time for each time sample. The selection is affected by the strength of the accent signal at the previous beat position, and the difference to the ideal beat interval. The indices of best previous beats are stored for each time sample, and in the end the single best sequence is obtained by backtracking through the previous beat records. For more details see [12].

2.3. Feature calculation

Next, beat synchronous MFCC and chroma features are calculated. Analysis frames are synchronized to start at a beat time and end before the next beat time, and one feature vector for each beat is obtained as the average of feature values during that beat. Beat synchronous frame segmentation has been used earlier e.g. in [7]. It has two main advantages: it makes the system insensitive to tempo changes between different chorus performances, and significantly reduces the size of the self-distance matrices and thus computational load. Prior to the analysis, the input signal is downsampled to 22050 kHz sampling rate.

The MFCC features are calculated in 30 ms hamming windowed frames during each beat, and the average of 12 MFCC features (ignoring the zeroth coefficient) for each beat is stored. We use 36 frequency bands spaced evenly on the mel-frequency scale, and the filters span the frequency range from 30Hz to the nyquist frequency. Chroma features are calculated in longer, 186 ms frames to get a sufficient frequency resolution for the lower notes. In our implementation, each bin of the discrete Fourier transform is mapped to exactly one of the twelve pitch classes C, C#, D, D#, E, F, F#, G, G#, A, A#, B, with no overlap. The energy is calculated from a range of six octaves from C3 to B8 and summed to the corresponding pitch classes. The chroma vectors are normalized by dividing each vector by its maximum value.

After the analysis, each inter-beat interval is represented with a MFCC vector and chroma vector, both of which are 12-dimensional.

2.4. Distance matrix calculation

The next step is to calculate the self-distance matrix (SDM) for the signal. Each entry $D(i, j)$ in the distance matrix indicates the distance of the music signal at time i to itself at time j. As we are using beat synchronous features, time is measured in beat units. Two distance matrices are used, one for the MFCC features and one for the chroma features. The entry $D_{mfcc}(i, j)$ of the MFCC distance matrix is calculated as the Euclidean distance of MFCC vectors of beats i and j. Correspondingly, in the chroma distance matrix $D_{chroma}(i, j)$ each entry corresponds to the Euclidean distance of the chroma vectors of beats i and j. Figures 2 and 3 show examples of a chroma and MFCC distance matrices, respectively. As the Euclidean distance is symmetric, the distance matrix will also be symmetric. Thus, the following operations consider only the lower triangular part of the distance matrix.

Alternatives to calculating two different distance matrices would be to concatenate the features before calculating the distances, or combine the features in the distance calculation step. The benefit of keeping the distance matrices separate is that different enhancement operations can be applied to the chroma and MFCC matrices. Based on our experiments, it seems beneficial to apply an enhancement only for the chroma distance matrix and not for the MFCC distance matrix. When long chords or notes are played during several adjacent beats, the chroma distance matrix will exhibit a square area of small distance values. An enhancement operation similar to the one described in [8] was found to be beneficial in removing these. The MFCC distance matrix does not exhibit similar areas as the MFCC features are insensitive to pitch information, so this would explain the MFCC distance matrix does not benefit from the enhancement. Moreover, summing the distance matrices first and then enhancing the summed matrix did not perform as well as enhancing the chroma matrix only and then summing with the MFCC matrix. The next section describes the used enhancement and SDM summing steps.

2.5. Enhancing and summing the distance matrices

Ideally, the distance matrix should contain diagonal stripes of small distance values at positions corresponding to repetitions of the chorus or refrain section. However, due to variations in the performance of the chorus at different times (articulation, improvisation, changing instrumentation), the diagonal stripes are often not very well pronounced. In addition, there may be additional small distance regions which do not correspond to chorus sections. To make diagonal segments of small distance values more pronounced in the distance matrix, an enhancement method similar to the one presented in [8] is utilized.

The chroma distance matrix $D_{chroma}(i, j)$ is processed with a 5 by 5 kernel. For each point (i, j) in the chroma distance matrix, the kernel is centred to the point (i, j). Six directional local mean values are calculated along the upper-left, lower-right, right, left, upper, and lower dimensions of the kernel. If either of the means along the diagonal is the minimum of the local mean values, the point (i, j) in the distance matrix is emphasized by adding the minimum value. If some of the mean values along the horizontal or vertical directions is the smallest, it is assumed that the value at (i, j) is noisy and it is suppressed by adding the largest of the local mean values. After the enhancement the diagonal lines corresponding to repeating sections are more pronounced.

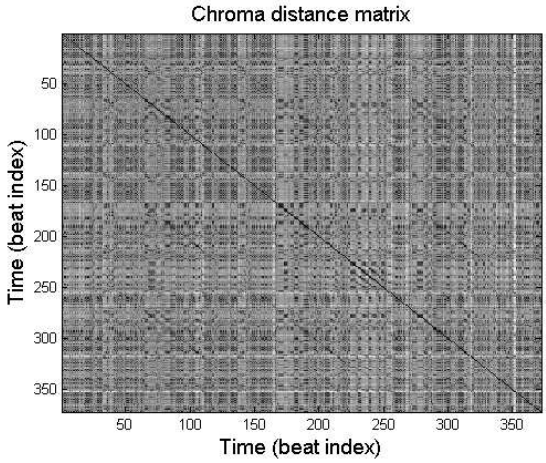


Figure 2: *The chroma distance matrix $D_{chroma}(i, j)$ of the song "Like a virgin" by Madonna.*

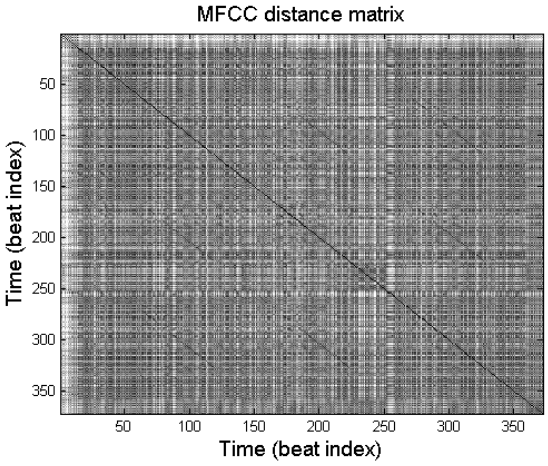


Figure 3: *The MFCC (timbre) distance matrix $D_{mfcc}(i, j)$ of the song "Like a virgin" by Madonna.*

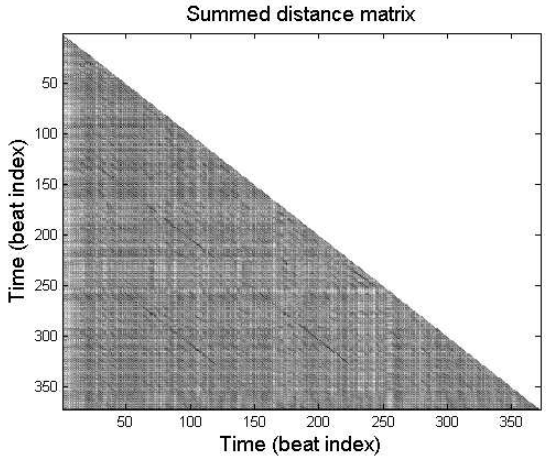


Figure 4. *The final distance matrix $D(i, j)$ of the song "Like a virgin" by Madonna obtained after summing the enhanced chroma distance matrix and MFCC distance matrix.*

After the enhancement step the chroma and MFCC distance matrices are summed. This gives the final distance matrix D, where the entries $D(i,j) = \widetilde{D}_{chroma}(i,j) + D_{mfcc}(i,j)$, where $\widetilde{D}_{chroma}$ is the chroma distance matrix after the above described enhancement operation. Figure 4 shows the summed distance matrix for Madonna's "Like a virgin". Weighted summation was also attempted for the different matrices with certain weight combinations, but equal weights (i.e. no weighting) seem to perform well. A slightly related approach to our distance matrix summing was presented by Marolt [13]. He constructed several beat synchronous melodic representations by comparing excerpts of different length, and then combined the representations by pointwise multiplication. This was reported to help in reducing noise in the self-similarity representation.

2.6. Detecting repetitions from the self-distance matrix

The following step attempts to detect which parts of the distance matrix correspond to a repetitive segment and which do not. The binarization method used here is similar to the one presented by Goto in [8], except that we operate on the low-triangular part of a distance matrix whereas Goto operated on the time-lag triangle. In addition, the filtering operations are simplified here and the threshold selection operations differ slightly.

When a sum is calculated along a diagonal segment of the distance matrix, a smaller value indicates a larger likelihood that the particular diagonal contains one or more line segments with small similarity values. A sum is calculated along the low-left diagonals k of the distance matrix, giving the values

$$F(k) = \frac{1}{M-k}\sum_{c=1}^{M-k} D(c+k,c)\ ,\ k = 1,...,M-1 \qquad (1)$$

where M is the number of beats in the song. Thus, $F(1)$ corresponds to the first diagonal below the main, $F(2)$ to the second below the main diagonal, and so on. The values of k corresponding to the smallest values of $F(k)$ indicate diagonals which are likely to have repetitions in them. With Eq. 1 there exists a possibility that some small-distance values are masked by high distance values that happen to locate at the same diagonal. Thus, it might be worth studying whether special methods to remove the effect of high-distance values would improve the performance. However, this was left for future research as the simple summing seems to work well.

A certain number of diagonals corresponding to minima in $F(k)$ are then selected. Before looking for minima in $F(k)$, it is "detrended" to remove cumulative noise from it. This is done by calculating a lowpass filtered version of $F(k)$, using a FIR lowpass filter with 50 taps, the value of each coefficient being 1/50. The lowpass filtered version of $F(k)$ is subtracted from $F(k)$.

The minima correspond to zero-crossings in the differential of $F(k)$. The smoothed differential of $F(k)$ is calculated by filtering $F(k)$ with an FIR filter having the coefficients $b_1(i) = K\text{-}i$, $i = 0, \ldots, 2K$, with $K = 1$. The minima candidates are obtained by finding the points where the smoothed differential of $F(k)$ changes its sign from negative to positive. The values of the minima are dichotomized into two classes with the Otsu method presented in [14], and the values smaller than the threshold are selected. We observed that sometimes it may happen that only a few negative peaks are selected using this threshold. This would mean that the following binarization would examine only a few diagonals of the distance matrix, increasing the possibility that some essential diagonal stripes are left unnoticed. To overcome this, we raise the threshold gradually until at least 10 minima (and thus diagonals) are selected. The subset of indices selected from all the diagonal indices $k \in [1, M-1]$ to search for line segments is denoted by Y.

The diagonals of the SDM selected for the line segment search are denoted by

$$g_y(c) = D(c+y,c)\ ,\ c = 1,...,M-y \qquad (2)$$

where $y \in Y$. The diagonals $g_y(c)$ of the distance matrix are smoothed by filtering with a FIR with coefficients $b_2(i) = ¼$, $i = 1, \ldots, 4$. Goto ([8]) performed another threshold selection with the Otsu method ([14]) to select a threshold to be used for detecting the line segments from the diagonals. However, we found it better to define a threshold such that 20% of the values of the smoothed diagonals $\widetilde{g}_y(c)$ are left below it, and thus 20% of values are set to correspond to diagonal repetitive segments. This threshold is obtained in a straightforward manner by concatenating all the values of $\widetilde{g}_y(c)$, $c = 1,...,M-y$ and $y \in Y$ into a long vector, sorting the vector, and selecting the value such that 20 % of the values are smaller. Points where $\widetilde{g}_y(c)$ exceeds the threshold are then set to one, others are set to zero. This gives the binarized distance matrix.

Next the binarized matrix is enhanced, such that diagonal segments where most values are ones (i.e. detected small distance segments) are enhanced to be all ones under certain conditions. This is done in order to remove gaps of few beats in such diagonal segments that are long enough. These kinds of gaps occur if there is a point of high distance within a diagonal segment (due to e.g. a variation in the musicians' performance). The enhancement process processes the binarized distance matrix with a kernel of length 25 (beats). Thus, at the position (i, j) of the binarized distance matrix B(i, j), the kernel analyzes the diagonal segment from $B(i, j)$ to $B(i$+25-1, j+25-1). If at least 65 % of the values of the diagonal segment are ones, $B(i, j) = 1$ and either $B(i$+25-2, j+25-2) = 1 or $B(i$+25-1, j+25-1) = 1, all the values in the segment are set to one. This removes short gaps in the diagonal segments. The length of the kernel is a parameter to the system, the value 25 was found to work well. Goto ([8]) did not report a need for such an enhancement process but we found it necessary.

2.7. Locating interesting segments

The result of the previous steps is an enhanced binarized matrix $B_e(i, j)$ where the value one indicates that that point corresponds to a repetitive section and zero corresponds that there is no repetition at that point. The next step is to find diagonal segments that are interesting, i.e. likely correspond to a chorus.

There may be repetitions that are too short to correspond to a chorus, such as those that occur e.g. when the same pattern of notes are repeatedly played with some instrument. By default, segments longer than four seconds are searched and used for further processing. In the case no segments longer than four seconds are found, the system tries to extend the segments until at least some segments longer than four seconds are detected. If this does not help, the length limit is relaxed and all segments are used.

With some songs there may be a very large number of repetitive diagonal segments at this point. Therefore, some of the segments are removed. For each diagonal segment found in the binarized matrix, the method looks for diagonal segments which are located close to it. Let us denote a diagonal segment which starts at (i, j) and ends at (i', j') with $\underline{x}_p = [i, j, i', j']$. Furthermore, the length $\Delta(\underline{x}_p) = j' - j + 1$ is the duration of the segment in beats. Given two segments $\underline{x}_1$ and $\underline{x}_2$, the segment $\underline{x}_2$ is defined to be close to $\underline{x}_1$ iff

$\underline{x}_2(1) \geq (\underline{x}_1(1) - 5)$ and $\underline{x}_2(3) \leq (\underline{x}_1(3) + 20)$ and

$|\underline{x}_2(2) - \underline{x}_1(2)| \leq 20$ and $\underline{x}_2(4) \leq (\underline{x}_1(4) + 5)$

where |.| denotes absolute value. The parameters were obtained by experimentation and may be changed.

For each segment, the method then lists its close segments fulfilling the conditions above, finds the segments that have more than three close segments, and removes the extra segments. If some segment with more than three close segments is in the removal list of some other segment, then it is not removed. The result of this step is a collection of the diagonal segments $\underline{x}_p$, $p = 1,...,P$ in the binarized matrix.

2.8. Selecting the diagonal segment most likely corresponding to a chorus

Next the method selects the segment most likely corresponding to a chorus. This is done by utilizing a novel heuristic scoring scheme which considers aspects such as the position of a repetition in the self distance matrix, the position of a repetition in relation to other repetitions in the SDM, average energy and average distance in the SDM during the repetition, and number of times the repetition occurs in the musical data.

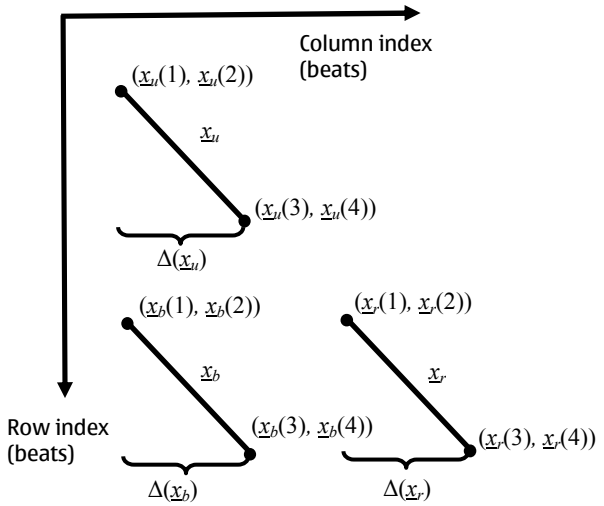


Figure 5: *Notations when giving scores to a group of three diagonal segments (detected stripes of small distance of the distance matrix). The units are measured in beats.*

2.8.1. Position of a repetition in the distance matrix

The first criterion used in making the decision is how close a diagonal segment is to an expected chorus position in the song. This is based on the observation that often in pop music there is a chorus at time corresponding to approximately one quarter of the song length. A partial score $s_1(\underline{x}_p)$ measures the difference of the middle column of segment $\underline{x}_p = [i, j, i', j']$ to one quarter of the song length:

$$s_1(\underline{x}_p) = 1 - \frac{\left|(j + \Delta(\underline{x}_p)/2) - round(M/4)\right|}{round(M/4)}, \quad (3)$$

where *M* is the length of the song in beats. The partial score $s_2(\underline{x}_p)$ measures the difference of the middle row of segment $\underline{x}_p$ to three quarters of the song length:

$$s_2(\underline{x}_p) = 1 - \frac{\left|(i + \Delta(\underline{x}_p)/2) - round(3 \cdot M/4)\right|}{round(M/4)}. \quad (4)$$

With $s_1(\underline{x}_p)$ and $s_2(\underline{x}_p)$ we give more weight to such segments that are close to the position of the diagonal stripe on the low left hand corner of Figure 4, which corresponds to the first occurrence of a chorus (and match to the third occurrence) and is often the most prototypically performed chorus, i.e. no articulation or expression.

2.8.2. Adjustment in relation to other repetitions

The second criterion relates to the adjustment of a segment within the distance matrix in relation to other repetitions. Motivated by the approach presented in [5], we look for possible groups of three diagonal stripes that might correspond to three repetitions of the chorus. See Figure 5 for an example of an ideal case. The search for possible groups of three stripes is done as follows: the method goes through each found diagonal segment $\underline{x}_u$, and looks for possible diagonal segments below it. If a segment below $\underline{x}_b$, $b \neq u$, is found, it looks for a segment $\underline{x}_r$ $r \neq u$, $r \neq b$, on the right from the segment $\underline{x}_b$. In order to qualify as a below segment, we require that $\underline{x}_b(1) > \underline{x}_u(3)$, and that there must be some overlap between the column indices of $\underline{x}_u$ and $\underline{x}_b$. To qualify as a right segment $\underline{x}_r$, there must be some overlap between the row indices of segments $\underline{x}_b$ and $\underline{x}_r$. The groups of three segments fulfilling the above criteria are denoted with $\underline{m}_z = [u, b, r]$, $z = 1,...,Z$. In theory there could be at maximum of $P(P-1)(P-2)$ such groups of three segments, in practice the number is much less. An arbitrary segment may belong to zero or several groups.

The groups of three stripes are then scored based on how close to ideal the group of three stripes is. This scoring affects the scores of some of the segments belonging to these groups. Four partial scores are calculated to measure the quality of each group of three stripes $\underline{m}_z = [u, b, r]$. The first partial score measures how close is the end point of the above segment $\underline{x}_u$ and below segment $\underline{x}_b$:

$$\sigma_1(z) = 1 - 2\left|\underline{x}_u(4) - \underline{x}_b(4)\right| / (\Delta(\underline{x}_b) + \Delta(\underline{x}_u)), \quad (5)$$

where $\underline{x}_u(4)$ and $\underline{x}_b(4)$ are the column indices of the end points of upper and below segments, respectively. The second partial score depends on the vertical alignment of upper and below segments:

$$\sigma_2(z) = \begin{cases} 1-(\underline{x}_u(2)-\underline{x}_b(2))/\Delta(\underline{x}_b) & \text{if } \underline{x}_b(2) < \underline{x}_u(2) \\ 1-(\underline{x}_b(2)-\underline{x}_u(4))/\Delta(\underline{x}_b) & \text{if } \underline{x}_b(2) > \underline{x}_u(4) \\ 1 & \text{otherwise} \end{cases} \quad (6)$$

The next score measures whether the segments $\underline{x}_b$ and $\underline{x}_r$ are of equal length:

$$\sigma_3(z) = 1 - \left|\Delta(\underline{x}_r) - \Delta(\underline{x}_b)\right| / \Delta(\underline{x}_b) . \quad (7)$$

The final partial score depends on the difference in the position of left and right segments:

$$\sigma_4(z) = 1 - \frac{2 \cdot \min(\left|\underline{x}_b(1)-\underline{x}_r(1)\right|, \left|\underline{x}_b(3)-\underline{x}_r(3)\right|)}{\Delta(\underline{x}_b)+\Delta(\underline{x}_r)}, \quad (8)$$

where 'min' denotes minimum operator.

The final score for the group of three segments $\underline{m}_z = [u,b,r]$ is the average of $\sigma_1(z)$, $\sigma_2(z)$, $\sigma_3(z)$, and $\sigma_4(z)$ denoted $\hat{\sigma}(z)$. Since this score considers a segment group, we need to decide whether all the segments in the group receive a score, or whether only certain segments. It was found beneficial to give the score to segment $\underline{x}_b$. The score could also be given to segment $\underline{x}_u$ as it may also correspond to the first instance of the chorus. However, the diagonal stripe corresponding to $\underline{x}_u$ is often longer than the actual chorus, it often consist e.g. of the repeating verse and chorus. It was observed that it gives better results to score the segment $\underline{x}_b$ as its length often more closely corresponds to the correct chorus length. Thus, depending on whether each found segment belong to at least one group of three segments, it will receive a score $s_3(\underline{x}_p) = \max \hat{\sigma}(y)$, $\{y \mid \underline{m}_y(2) = p\}$. The maximum is taken as each segment may belong to more than one group. If a segment $\underline{x}_p$ does not belong to any group of three segments, $s_3(\underline{x}_p) = 0$.

2.8.3. Average energy and distance of a segment

The next criterion $s_4(\underline{x}_p)$ is the average logarithmic energy of the portion of the music signal defined by the column indices of segment $\underline{x}_p$ normalized with the average energy over the whole signal. Using the energy as one criterion gives more weight to such segments that have high average energy, which is often a characteristic of chorus sections. The partial score $s_5(\underline{x}_p)$ takes into account the average distance value during the segment: the smaller the distance during the whole segment the more likely it is that the segment corresponds to a chorus:

$$s_5(\underline{x}_p) = 1 - \phi(\underline{x}_p)/\Phi , \quad (9)$$

where $\phi(\underline{x}_p)$ is the median distance value of the diagonal segment $\underline{x}_p$ in the distance matrix, and Φ is the average distance value over the whole distance matrix.

2.8.4. Number of times the repetition occurs

The last partial score $s_6(\underline{x}_p)$ considers the number of times the repetition occurs. Other diagonal segments locating on top of or below segment $\underline{x}_p$ are indications that the segment defined by the column indices of $\underline{x}_p$ is repeating more than once. To get a score for this, a search is done for all segment candidates $\underline{x}_p$, and a count is made of all those other segments $\underline{x}_q$ which fulfill the condition $\left|\underline{x}_p(2)-\underline{x}_q(2)\right| \le 0.2 \cdot \Delta(\underline{x}_q)$ and $\left|\underline{x}_p(4)-\underline{x}_q(4)\right| \le 0.2 \cdot \Delta(\underline{x}_q)$. The count of other segments $\underline{x}_q$ fulfilling the above criterion is stored as the score for all segment candidates $\underline{x}_p$. When these counts for all segment candidates have been obtained, the values are normalized by dividing with the maximum count, giving the final values for a score $s_6(\underline{x}_p)$ for each segment.

2.8.5. Selecting the most likely chorus segment

The remaining task is to select the most likely chorus segment based on the various criteria. For each segment $\underline{x}_p$, a score is given as

$$\begin{aligned} S(\underline{x}_p) &= 0.5 \cdot s_1(\underline{x}_p) + 0.5 \cdot s_2(\underline{x}_p) \\ &+ s_3(\underline{x}_p) + 0.5 \cdot s_4(\underline{x}_p) + s_5(\underline{x}_p) + 0.5 \cdot s_6(\underline{x}_p). \end{aligned} \quad (10)$$

There is a possibility to optimize the weights in Eq. 10, which we did not fully explore in the fear of over fitting data but just manually selected weights that gave good performance on a small set of music files. The segment $\underline{x}_p$ maximizing the score *S* is selected as the most likely chorus segment. If at least one group of three diagonal stripes fulfilling the criteria of section 2.8.2 has been found, the choice is made among such segments $\underline{x}_o$ for which $s_3(\underline{x}_o) \neq 0$, i.e. those that have been at an appropriate position in at least one group of three diagonal stripes. If no sets of three stripes is found, the selection is made among all the segments by maximizing *S*. In this case the group score $s_3(\underline{x}_p) = 0$ for all segment candidates. The result of this step is an initial chorus segment $\underline{x}_c$.

2.9. Finding the exact location of the chorus

Next the exact location and length of the chorus section is refined using filtering in two or one dimensions. 2D kernels have earlier been used by Shiu et al. to analyze local similarity of the signal by detecting repeated chord successions from a measure-level self-similarity matrix [15]. Here, we use 2D filters to get the exact position for a chorus segment. Often, the time signature in western pop and rock music has a 4/4 time signature, and the length of a chorus section is 8 or 16 measures (32 or 64 beats, respectively) [9]. In addition, the chorus may consist of two repeating subsections of equal length. Two dimensional filter kernels are constructed to model the pattern of ideal small-distance stripes that would be caused by a chorus of 8 or 16 measures long, with two repeating subsections. Figure 6 shows the filter of dimension 32 by 32 beats, with two 16 by 16 beats long repeating subsections. This is the idealized shape of the small-distance stripes occurring in the distance matrix if the song has this kind of chorus. The second filter is

similar but of dimension 64 by 64, and with diagonals modeling the 32 beat long subsections.

The area of the distance matrix surrounding the chorus candidate is filtered with these two kernels. The chorus candidate start column is denoted with $\underline{x}_c(2)$ and the end column $\underline{x}_c(4)$. The columns of the low triangular distance matrix starting from $\max(1, \underline{x}_c(2) - N_f/2)$ to $\min(\underline{x}_c(4) + N_f/2, M)$ are selected as the area from which to search for the chorus. N_f is the dimension of the filter kernel, either 32 or 64, and M is the length of the song in beats. min and max are applied to prevent over indexing. If the length of the area above in the column dimension is shorter than the filter dimension, this step is omitted. The area is limited to lessen the computational load and to prevent the refined chorus section from departing too much from the initial chorus candidate.

When the upper left-hand side corner of the filter with dimension N_f is positioned in (i, j) at the distance matrix, the following values are calculated: mean distance $\alpha(i, j, N_f)$ along the diagonals (marked with black color in Figure 6), mean distance $\beta(i, j, N_f)$ along the main diagonal and mean distance $\lambda(i, j, N_f)$ of the surrounding area (white color in Figure 6). The ratio $\rho_\alpha(i, j, N_f) = \alpha(i, j, N_f) / \lambda(i, j, N_f)$ indicates how well the position matches with a chorus with two identical repeating subsections, and the ratio $\rho_\beta(i, j, N_f) = \beta(i, j, N_f) / \lambda(i, j, N_f)$ how well the position matches a strong repeating section of length N_f with no subsections. The smaller the ratio, the smaller the values on the diagonal compared to the surrounding area. The smallest value of $\rho_\alpha(i, j, N_f)$ and $\rho_\beta(i, j, N_f)$ and the corresponding indices are stored for both filters, i.e. with N_f=32 and N_f =64. These smallest values are denoted by $\rho'_\alpha(N_f)$ and $\rho'_\beta(N_f)$.

Several heuristics are then used to select the final chorus position and length based on the filtering results, or if the conditions are not met then another filtering in one dimension along the initial chorus segment is performed. The final chorus section is selected according to the two dimensional filtering, if the smallest ratios are small enough. The following heuristics are used, although many other alternatives would be possible. These rules below have been obtained via trial and error by experimenting with a subset of 50 songs from our music collection.

If $\rho'_\alpha(64) < \rho'_\alpha(32)$, it indicates a good match with the 64 beat long chorus with two 32 beat long subsections. The chorus starting point is selected according to the column index of the point which minimized $\rho'_\alpha(64)$, and its length is taken as 64 beats. Else, if the length of the initial chorus section is less than 32, the chorus section is adjusted according to the point minimizing $\rho'_\alpha(32)$ only if the chorus beginning would change at maximum one beat from the initial location. Finally, if the length of the initial chorus section is closer to 48 than 32 or 64 and $\rho'_\alpha(32) < \rho'_\alpha(64)$ and $\rho'_\beta(32) < \rho'_\beta(64)$ and the column index of the point minimizing $\rho'_\alpha(32)$ is the same as the point minimizing $\rho'_\beta(32)$, the chorus is

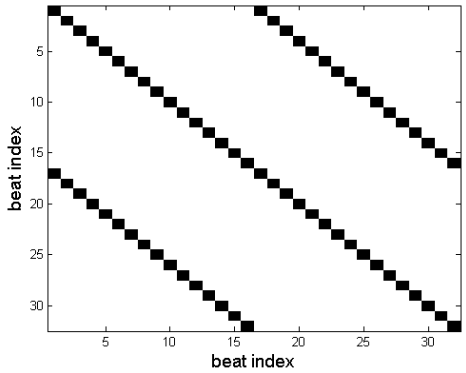


Figure 6. *Two dimensional filter kernel modelling the stripes occurring if the song has a chorus of 32 beats in length with two 16 beat repeating subsections. The average distance is calculated along the entries marked with black colour, and compared to the average distance of locations corresponding to rest of the kernel (white entries).*

set to start at the point minimizing both $\rho'_\alpha(32)$ and $\rho'_\beta(32)$ and its length is set to 32. Thus, these are special rules to adjust the chorus section if it seems likely that there song has either a 32 or 64 beats long chorus with identical subsections half its size.

In many cases, the above conditions are not met, and the chorus section is adjusted by performing filtering along the diagonal values of the initial chorus section and a small offset of five beats before and after its beginning and end. Thus, if the row and column indices of the initial chorus section are denoted with $(\underline{x}_c(1), \underline{x}_c(2))$ (the beginning) and $(\underline{x}_c(3), \underline{x}_c(4))$ (the end), the values to be filtered are found along the line from $(\underline{x}_c(1) - 5, \underline{x}_c(2) - 5)$ to $(\underline{x}_c(3) + 5, \underline{x}_c(4) + 5)$.

The filtering is done with two kernels of length 32 and 64, but now on one dimension along the diagonal distance values of the initial chorus section and its immediate surroundings. The ratio $r(32)$ is the smallest ratio of mean of distance values on the 32 point kernel to the values outside the kernel. If $r(32) < 0.7$ and the length of the initial chorus section is closer to 32 than 64, the chorus starting point is set according to the location minimizing $r(32)$ and its length is set to 32. If the length of the initial chorus section is larger than 48, the final chorus start location and length is selected according to the one giving the smaller score. This step in our method looks for the best position of the chorus section e.g. in the case the diagonal stripe selected as the chorus section consists of a longer repetition of a verse and chorus, for example. Note that the method is not limited to 4/4 time signature and chorus lengths of 32 or 64: if the conditions above are not met, the chorus section is kept as the one returned from the binarization process. In these cases its length does not have to be 32 or 64.

3. EVALUATION

The method was evaluated on database consisting of 206 popular and rock music pieces. Most of the pieces have a clear verse-chorus structure, although there are some instances where the structure is

less obvious. The chorus sections were annotated manually from the pieces. The annotations were made with a dedicated tool, which showed the beat synchronized SDM of the signal aligned with the signal itself. The self-distance matrix visualization significantly speeded up the annotation work as the different sections were more easily found.

Performance of the system is measured with the F-measure, defined as the harmonic mean of the recall rate (R) and precision rate (P): $F = (2RP) / (R + P)$. To calculate R and P, we find the annotated chorus section with maximum overlap with the detected chorus section, and calculate the length l_{corr} of the section where the detected chorus section overlaps with the annotated section. R is calculated as the ratio l_{corr} to the length of the annotated chorus section, and P is the ratio of l_{corr} to the length of the detected chorus section. The F-measure is calculated for each track, and the reported overall F-measure is the average of the F-measures over all tracks.

Table 1 shows the chorus detection results. Baseline is the normal system. The most common error is small offsets in the beginning and/or end locations of the chorus section that reduce the score. The second row represents the results when the output chorus section length is fixed to 30 seconds. Being able to output a fixed length segment may be desirable in some applications, such as music preview. If the initial chorus section is shorter than 30 seconds, expanding is done by following the diagonal chorus segment into the direction of minimum distance in the SDM. Correspondingly, shortening is done by dropping in turn the point with larger distance value from either end. As the recall rate increases when the 30 s limit is applied, the method has not always captured the whole chorus section. If it is desirable that the thumbnail section captures the chorus and it's acceptable if the section extends beyond the chorus, the 30s option can be used. The method is efficient; it takes about ten seconds to process a song with an average duration of three to four minutes on a Windows XP computer with a 2.8 GHz Intel Xeon processor.

Method	P	R	F
Baseline	89%	83%	86%
30s length	70%	92%	79%

Table 1: *Chorus detection results.*

4. CONCLUSIONS

A method for chorus detection from popular and rock music was presented. The method utilizes a novel feature analysis front-end where the MFCC and chroma distance matrices are summed and a two step procedure of initial chorus selection and section refinement. A novel heuristic scoring scheme was proposed to select the initial chorus candidate from the binarized distance matrix, and a novel approach utilizing image processing filters is used to refine the final position and length of the chorus candidate. Evaluations on a manually annotated database of 206 songs demonstrate that the performance of the method is sufficient for practical applications, such as previewing playlists of popular and rock music. Moreover, the method is computationally efficient.

5. REFERENCES

[1] G. Peeters, A. La Burthe, X. Rodet, "Toward Automatic Music Audio Summary Generation from Signal Analysis", in *Proc. of the 3rd International Conference on Music Information Retrieval, ISMIR 2002*, Paris (France), October 2002.

[2] B. Logan, S. Chu, "Music summarization using key phrases," in *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, *ICASSP 2000*, vol. 2, pp. 749-752, Istanbul, Turkey, May 2000.

[3] M. Levy, M. Sandler, M. Casey, "Extraction of High-Level Musical Structure From Audio Data and Its Application to Thumbnail Generation," in *Proc. IEEE ICASSP 2006*, vol. V, pp. 13-16.

[4] C. Rhodes, Casey, S. Abdallah, M. Sandler, "A Markov-chain monte-carlo approach to musical audio segmentation," in *Proc. IEEE ICASSP 2006*, vol. V, pp. 797-800.

[5] J. Wellhausen and H. Crysandt, "Temporal Audio Segmentation Using MPEG-7 Descriptors," in *Proc. of the SPIE International Symposium on ITCom 2003 - Internet Multimedia Management Systems IV,* Orlando (FL), USA, September 2003.

[6] M. Cooper, J. Foote, "Summarizing Popular Music Via Structural Similarity Analysis," in *Proc. of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, WASPAA 2003*, October 19-22, 2003, New Paltz, NY.

[7] M. A. Bartsch, G. H. Wakefield, "Audio Thumbnailing of Popular Music Using Chroma-Based Representation," *IEEE Trans. on Multimedia*, vol. 7, no. 1, Feb. 2005, pp. 96-104.

[8] M. Goto: "A Chorus Section Detection Method for Musical Audio Signals and Its Application to a Music Listening Station," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 14, no. 5, Sept. 2006 pp. 1783 – 1794.

[9] N. Maddage, "Automatic Structure Detection for Popular Music," *IEEE Multimedia*, Jan.-March 2006, vol. 13, no. 1, pp. 65-77.

[10] J. Paulus, A. Klapuri, "Music Structure Analysis by Finding Repeated Parts", in *Proc. of the 1st Audio and Music Computing for Multimedia Workshop (AMCMM2006)*, Santa Barbara, California, USA, October 27, 2006, pp. 59-68.

[11] J. Seppänen, A. Eronen, and J. Hiipakka, "Joint Beat & Tatum Tracking from Music Signals", In *Proc. of the 7th International Conference on Music Information Retrieval, ISMIR 2006,* Victoria, Canada, 8 - 12 October 2006.

[12] D. Ellis, "Beat Tracking with Dynamic Programming", MIREX 2006 Audio Beat Tracking Contest system description, Sep 2006, available at http://www.ee.columbia.edu/~dpwe/pubs/Ellis06-beattrack.pdf

[13] M. Marolt, A Mid-level Melody-based Representation for Calculating Audio Similarity, In *Proc. of the 7th International Conference on Music Information Retrieval, ISMIR 2006,* Victoria, Canada, 8 - 12 October 2006.

[14] N. Otsu, A threshold selection method from gray-level histograms, *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, no. 1, pp. 62-66, Jan. 1979.

[15] Y. Shiu, H. Jeong, C.-C. Jay Kuo, "Similarity Matrix Processing for Music Structure Analysis", In *Proc. of the 1st Audio and Music Computing for Multimedia Workshop (AMCMM2006)*, October 27, 2006, Santa Barbara, California, USA.