Marcin Detyniecki
Ulrich Leiner
Andreas Nürnberger (Eds.)

Adaptive Multimedia Retrieval

Identifying, Summarizing, and Recommending Image and Music

LNCS 5811

6th International Workshop, AMR 2008
Berlin, Germany, June 2008
Revised Selected Papers

Springer

Lecture Notes in Computer Science 5811

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Marcin Detyniecki Ulrich Leiner
Andreas Nürnberger (Eds.)

Adaptive Multimedia Retrieval

Identifying, Summarizing, and Recommending Image and Music

6th International Workshop, AMR 2008
Berlin, Germany, June 26-27, 2008
Revised Selected Papers

Springer

Volume Editors

Marcin Detyniecki
Université Pierre et Marie Curie
Paris, France
E-mail: marcin.detyniecki@lip6.fr

Ulrich Leiner
Fraunhofer Institute for Telecommunications
Heinrich Hertz Institute
Berlin, Germany
E-mail: ulrich.leiner@hhi.fraunhofer.de

Andreas Nürnberger
Otto-von-Guericke University Magdeburg
Magdeburg, Germany
E-mail: andreas.nuernberger@ovgu.de

Library of Congress Control Number: 2010931218

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

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

ISSN 0302-9743
ISBN-10 3-642-14757-7 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-14757-9 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

This book is a selection of the revised contributions that were initially submitted to the International Workshop on Adaptive Multimedia Retrieval (AMR 2008). The workshop was organized at the Fraunhofer Institute for Telecommunications HHI, Berlin, Germany, during June 26–27, 2008.

The goal of the AMR workshops is to intensify the exchange of ideas between different research communities, to provide an overview of current activities in this area and to point out connections between the diverse researches communities, in particular the ones focussing on multimedia retrieval and artificial intelligence. In this spirit, the first three events were collocated with Artificial Intelligence conferences: in 2003 as a workshop of the 26th German Conference on Artificial Intelligence (KI 2003); in 2004 as part of the 16th European Conference on Artificial Intelligence (ECAI 2004) and in 2005 as part of the 19th International Joint Conference on Artificial Intelligence (IJCAI 05). Because of its success, in 2006 the University of Geneva, Switzerland organized the workshop for the first time as a standalone event. The motivation of the participants led us to continue this path, and thus AMR 2007 and AMR 2008 were again organized as independent events at the Laboratoire d'Informatique de Paris VI in France and at the Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute (HHI) in Berlin, respectively.

The workshop in 2008 revealed four main subtopics: summarization, identification and recommendation. These challenges addressed image, Web and music data, the latter being a strong and new push in the AMR series. Therefore, in this edition, adaptive retrieval—the core subject—was tackled from quite different and innovative perspectives.

In order to address the problem of information overflow, the research community proposes to summarize the available information, by structuring or extracting relevant data. The reduction can take several forms and granularities, for instance in the case of audio data the extraction of relevant musical thumbnails is an important topic, while in the visual domain the summarization of large image sets is the challenge. In both cases the aim is to discard the irrelevant while covering most of the available information.

When addressing adaptation the user remains in the center of attention and recommendation is the current trend. The discussions are centered around the question of what is the object to be recommended and what are its specific properties. In the case of music or images the objects tend to be the entire document and recommendation is usually based on user preferences. While in the case of Web applications, keywords were the target and semantic analysis the core approach.

A key challenge always addressed at AMR is how to tackle the semantic gap. Media-specific identification techniques were proposed and we witness two

general trends: tagging and tracking. The first tackles the problem of labelling general multimedia objects and the second explores the idea of recognizing almost exact copies, as for instance the picture of a painting in a museum.

We believe that the above trends are representative and thus this book provides a good and conclusive overview of the current research in this area.

Finally, we would like to thank all members of the Program Committee for supporting us in the reviewing process, the workshop participants for their willingness to revise and extend their papers for this book, the sponsor for their financial support and Alfred Hofmann from Springer for his support in the publishing process.

November 2009

Marcin Detyniecki
Ulrich Leiner
Andreas Nürnberger

Organization

Program Chairs

Marcin Detyniecki	CNRS, Laboratoire d'Informatique de Paris 6, France
Ulrich Leiner	Fraunhofer Institute for Telecommunications HHI, Berlin, Germany
Andreas Nürnberger	Otto-von-Guericke University, Magdeburg, Germany

Technical Chair

Sebastian Stober	Otto-von-Guericke University, Magdeburg, Germany

Local Organization

Christian Hentschel	Fraunhofer Institute for Telecommunications HHI, Berlin, Germany

Program Committee

Jenny Benois-Pineau	University of Bordeaux, LABRI, France
Stefano Berretti	Università di Firenze, Italy
Susanne Boll	University of Oldenburg, Germany
Eric Bruno	University of Geneva, Switzerland
Bogdan Gabrys	Bournemouth University, UK
Xian-Sheng Hua	Microsoft Research, Beijing, China
Philippe Joly	Université Paul Sabatier, Toulouse, France
Gareth Jones	Dublin City University, Ireland
Joemon Jose	University of Glasgow, UK
Stefanos Kollias	National Technical University of Athens, Greece
Stéphane Marchand-Maillet	University of Geneva, Switzerland
Trevor Martin	University of Bristol, UK
José María Martínez Sánchez	Universidad Autónoma de Madrid, Spain
Bernard Merialdo	Institut Eurécom, Sophia Antipolis, France
Jan Nesvadba	Philips Research, Eindhoven, The Netherlands
Gabriella Pasi	Università degli Studi di Milano Bicocca, Italy

Valery Petrushin	Accenture Technology Labs, Chicago, USA
Stefan Rüger	The Open University, Milton Keynes, UK
Simone Santini	Universidad Autonoma de Madrid, Spain
Raimondo Schettini	University of Milano Bicocca, Italy
Ingo Schmitt	University of Cottbus, Germany
Nicu Sebe	University of Amsterdam, The Netherlands
Alan F. Smeaton	Dublin City University, Ireland
Arjen De Vries	CWI, Amsterdam, The Netherlands

Supporting Institutions

Fraunhofer Institute for Telecommunications HHI, Berlin
Otto-von-Guericke University, Magdeburg, Germany
Universite Pierre & Marie Curie, Paris, France
Laboratoire d'Informatique de Paris 6 (LIP6), France

Table of Contents

Tagging and Structuring Image Collections

Systems for Still and Motion Images

The Future of Audio Reproduction

Technology – Formats – Applications

Matthias Geier[1], Sascha Spors[1], and Stefan Weinzierl[2]

[1] Deutsche Telekom Laboratories, Quality and Usability Lab, TU Berlin,
Ernst-Reuter-Platz 7, 10587 Berlin, Germany
{Matthias.Geier,Sascha.Spors}@telekom.de
http://www.qu.tu-berlin.de
[2] Audio Communication Group, TU Berlin,
Einsteinufer 17, 10587 Berlin, Germany
Stefan.Weinzierl@tu-berlin.de
http://www.ak.tu-berlin.de

Abstract. The introduction of new techniques for audio reproduction such as binaural technology, *Wave Field Synthesis* and *Higher Order Ambisonics* is accompanied by a paradigm shift from *channel-based* to *object-based* transmission and storage of spatial audio. The separate coding of source signal and source location is not only more efficient considering the number of channels used for reproduction by large loudspeaker arrays, it will also open up new options for a user-controlled soundfield design. The paper describes the technological change from stereophonic to array-based audio reproduction techniques and introduces a new proposal for the coding of spatial properties related to auditory objects.

1 Introduction

Audio recording, transmission and reproduction has been a very active field of research and development in the past decades. Techniques for the reproduction of audio signals including a respresentation of the spatial configuration and the acoustical environment of the corresponding sound sources have been an important aspect of recent innovations such as multichannel audio for cinema and DVD and new techniques for audio reproduction, which are primarily used in a research context so far.

Stereophonic reproduction is currently the most widespread audio reproduction technique. However, the spatial cues of an auditory scene, which allow the listener to localize sound sources and to identify features of the acoustical environment, are only preserved to a limited degree. This has led to a variety of new techniques for audio reproduction such as binaural technology, *Wave Field Synthesis* (WFS) and *Higher Order Ambisonics* (HOA). The introduction of these techniques is accompanied by a paradigm shift from *channel-based* to *object-based* transmission and storage of spatial audio features. The separate coding of source signal and source location is not only mandatory with respect to the high number of sometimes several hundred reproduction channels used for large loudspeaker

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 1–17, 2010.

arrays for WFS or HOA, it will also be the basis for interactive installations in which the user has access to the spatial properties of the reproduced soundfield and is able to adapt it to his individual requirements or aesthetic preferences.

This contribution discusses the technological change from stereophonic to advanced audio reproduction techniques, highlights the need for an object-based description of audio scenes and discusses formats for the coding of spatial properties related to auditory objects.

2 Channel-Based Audio Reproduction

The *stereophonic* approach to transmission and storage of spatial audio implies a multichannel reproduction system, from traditional two-channel stereophony to modern configurations with five, seven or even more loudspeakers, as they are used for cinema, home theater or – more recently – also for pure audio content. Based on an auditory illusion, the so-called *phantom source*, stereophony spans a panorama between pairs of loudspeakers, on which sound sources can appear. These virtual locations are hard-coded as signal relations between the two channels feeding the respective pair of loudspeakers. It can thus be considered a channel-based approach to spatial coding and transmission, as opposed to more recent, object-based approaches, where spatial information and audio signals are transmitted independently.

2.1 Psychoacoustics of Stereophony

Phantom sources emerge when pairs of loudspeakers produce largely identical signals exhibiting only a small time lag or a level difference between them. The listener will then perceive a virtual sound source located on the loudspeakers basis. The perceived location is determined by the time lag, by the level difference, or by a combination of both effects, whereby the effect of a level difference of 1 dB is approximately equivalent to a time lag of 60 μs (Fig. 1). The intended location will, however, only appear for a listener on the symmetry axis of the loudspeaker pair. A configuration with loudspeakers and listener on the corners of an equilateral triangle, yielding an aperture angle of 60°, is generally considered as ideal. Any deviation of this equidistant listener location will introduce additional time differences and thus offset the correct source positions as they are indicated in Fig. 1 (right).

It should be noted that the emergence of a phantom source is an auditory illusion generated by an artificial soundfield. In natural acoustic environments, nearly identical sound signals arriving from different directions of incidence do not exist. In this artificial situation, the auditory system obviously tries to find a natural model matching the perceived sensory information as closely as possible, hence suggesting a source location which would yield the same interaural time and/or level differences, which are known to be the most important cues for sound localization. The ear signals of a frontal phantom source and a frontal real sound source are, however, significantly different. Considering the four transfer

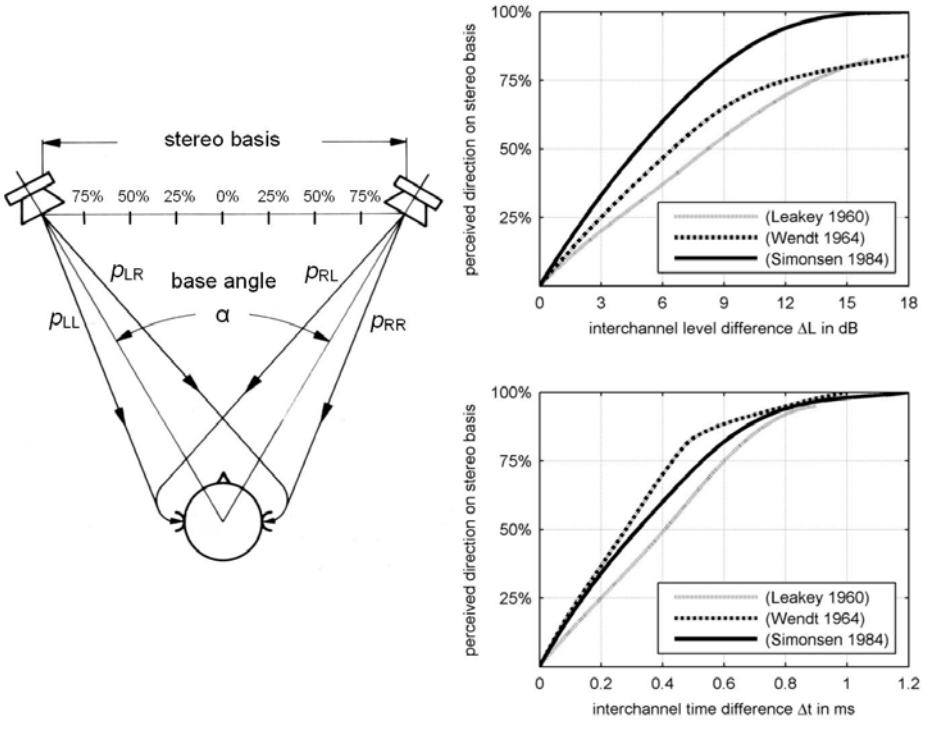


Fig. 1. *Left:* Loudspeaker configuration for two-channel stereophony with loudspeaker-ear transmission paths p_{XX}. The base angle α is usually chosen to be 60°. *Right:* Localisation of phantom sources on the loudspeaker basis against time and level differences between stereophonic signals. Values have been determined in listening tests with speech stimuli (Leakey [1], Simonsen, cited in [2]) and switched clicks (Wendt [3]).

paths from left/right loudspeaker to left/right ear ($p_{LL}, p_{LR}, p_{RL}, p_{RR}$, see Fig. 1, left) and the different source locations (±30° vs. 0° for a stereophonic vs. a real source), a considerable spectral difference can be expected, due to the comb filter caused by two consecutive signals at both ears (p_{LL} and p_{RL} for the left ear) and due to different effective *Head-Related Transfer Functions* (HRTFs) related to different angles of incidence. However, the perceived timbral distortion is much smaller than can be expected from a spectral analysis. A convincing explanation for this effect has still to be given. Theile has suggested that our auditory system might first determine a sorce location and then form a timbral impression after having compensated for the respective HRTF ([4], discussed by [5]), although there is no neurophysiological evidence for this hypothesis yet.

2.2 History and Formats

The formation of a fused auditory event generated by two signals of adjacent loudspeakers is the basis for spatial audio reproduction by all stereophonic

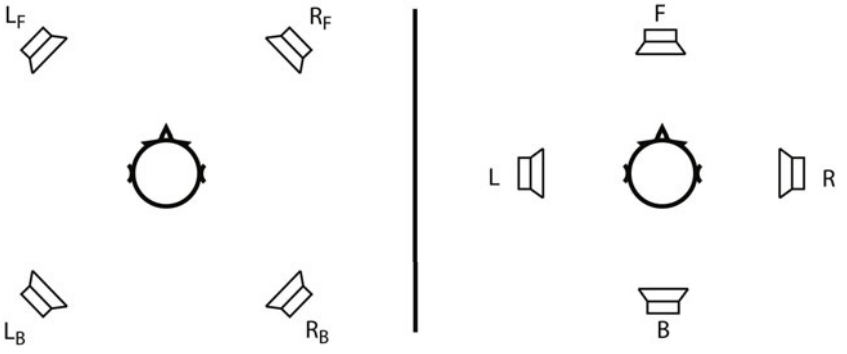


Fig. 2. Quadrophonic loudspeaker configurations: Scheiber array (left) and Dynaquad array (right)

reproduction systems. These include the classical two-channel stereophony which was studied at EMI already around 1930 [6], realised on two-channel magnetic tape from 1943 on in Germany [7], and distributed on stereo disc from 1958 on.

Quadrophonic loudspeaker configurations (Fig. 2) were first used in an experimental environment, for electronic and electroacoustic compositions such as *Symphonie pour un homme seul* (Pierre Schaeffer, 1951) or *Gesang der Jünglinge* (Karl-Heinz Stockhausen, 1956). The music industry's attempt to establish quadrophony as a successor to stereophony between 1971 and 1978 ultimately failed, due to the incompatibility of different technical solutions for 4-2-4 matrix systems which used conventional two-channel vinyl discs with additional encoding and decoding [8] offered by different manufacturers, as well as due to elementary psychoacoustic restrictions. No stable phantom sources can be established between the lateral pairs of loudspeakers [9], hence, the original promise of making the whole area of sound incidence around the listener accessible could not be fulfilled. In addition, while the sound source locations encoded in two-channel stereophony can be perceived largely correctly as long as the listener has an equal distance to the loudspeakers, a symmetrical listener position within all loudspeaker pairs of a quadrophonic loudspeaker array is only given in one central *sweet spot*.

In multichannel sound for cinema, where the spatial transmission of speech, music and environmental sounds has to be correct for a large audience area, these restrictions have been considered ever since the introduction of *Dolby Stereo* in 1976. The basic loudspeaker configuration for mix and reproduction has not changed since then (see Fig. 3). The center speaker (C) has always been used for dialog, thus providing a consistent central sound perception for the whole audience. The frontal stereo pair (L, R) is primarily used for music and sound effects. The surround speakers (S), distributed all over the walls around the audience area, are fed by a mono signal in the original *Dolby Stereo* format, while they are fed with two signals (*Left Surround, Right Surround*) or even three signals, including an additional *Back Surround*, in modern digital formats (Dolby Digital, DTS, SDDS).

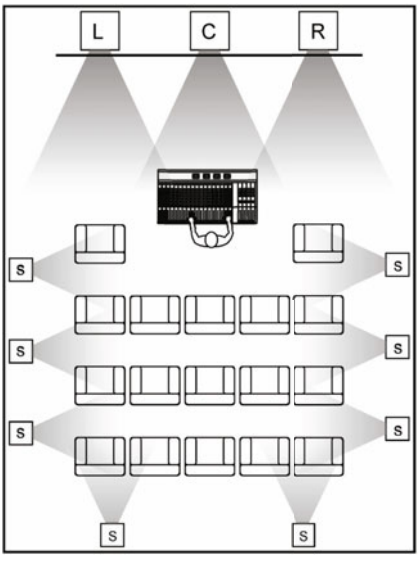


Fig. 3. Loudspeaker configuration for production and reproduction of 4.0, 5.1, 6.1 sound formats in cinema

3 Advanced Sound Spatialization Techniques

Stereophonic reproduction techniques are currently widespread in application areas like home entertainment and cinemas. However, these techniques exhibit a number of drawbacks, like e.g. the sweet spot, that have led to the development of advanced audio reproduction techniques. This section will give a brief overview on the major techniques, their physical background and their properties. Please refer to the cited literature for a more in-depth discussion of the particular methods.

3.1 Binaural Reproduction

Binaural reproduction aims at recreating the acoustic signals at the ears of a listener such that these are equal to a recorded or synthesized audio scene. Appropriately driven headphones are typically used for this purpose. The term *binaural reproduction* refers to various techniques following this basic concept. Audio reproduction using *Head-Related Transfer Functions* (HRTFs) is of special interest in the context of this contribution. The ability of the human auditory system to localize sound is based on exploiting the acoustical properties of the human body, especially the head and the outer ears [10]. These acoustical properties can be captured by HRTFs by measuring the transfer function between a sound source and the listener's ears. Potentially these measurements have to be undertaken for all possible listener positions and head poses. However, it is typically assumed that the listeners position is stationary. HRTF-based reproduction is then implemented by filtering a desired source signal with the appropriate HRTF

for the left and right ear. In order to cope for varying head orientations, head tracking has to be applied. Head-tracked binaural reproduction is also refereed to as dynamic binaural resynthesis [11].

Binaural reproduction has two major drawbacks: (1) it may not always be desired to wear headphones and (2) reproduction for large audiences or moving listeners is technically complex. The first drawback can be overcome, within limits, by using loudspeakers for binaural reproduction. In this case, appropriate crosstalk cancelation has to be employed in order that the signals at both ears of the listener can be controlled independently [12]. Such crosstalk cancelation typically exhibits a very pronounced sweet spot. Alternatives to binaural reproduction for potentially larger audiences will be introduced in the following. The first two are based on the physical reconstruction of a desired sound field within a given listening area.

3.2 Higher Order Ambisonics

Higher Order Ambisonics (HOA) and related techniques [13,14,15,16] base on the concept of single-layer potentials and theoretically provide a physically exact solution to the problem of sound field reproduction. The underlying theory assumes that a continuous single layer potential (secondary source distribution) surrounds the listening area. Appropriate weights (driving functions) applied to the secondary source distribution allow to reproduce almost any desired sound field within the listening area. Although arbitrary secondary source contours which enclose the receiver area are theoretically possible, explicit solutions are currently exclusively available for spherical and circular geometries.

The continuous distribution of secondary source is approximated in practice by a spatially discrete distribution of loudspeakers. This constitutes a spatial sampling process which may lead to spatial aliasing. The artifacts due to spatial sampling result in a pronounced artifact-free area in the center of the loudspeaker arrangement [17] that HOA and related techniques exhibit. The size of this area decreases with increasing frequency of the signal to be reproduced. For feasible loudspeaker setups the size of the artifact-free reproduction area is typically smaller than a human head at the upper end of the audible frequency range. Outside, spatial sampling artifacts arise that may be perceived as coloration of the desired sound field [18]. A number of HOA systems have been realized at various research institutes and other venues.

3.3 Wave Field Synthesis

Like HOA, *Wave Field Synthesis* (WFS) aims at physically recreating a desired sound field within a given listening area. However, the theoretical background of WFS differs in comparison to HOA [17]. WFS is based on the quantitative formulation of the *Huygens-Fresnel-Principle*, which states that a propagating wave front can be synthesized by a superposition of simple sources placed on the wave front.

WFS has initially been developed for linear secondary source distributions [19], where *Rayleigh integrals* describe the underlying physics. No explicit solution of

the reproduction problem is required in order to derive the secondary source driving function. The initial concept of WFS has been extended to arbitrary convex secondary source layouts [20] which may even only partly enclose the listening area. As for HOA, the continuous secondary source distribution is approximated by spatially discrete loudspeakers in practice. The resulting spatial sampling artifacts for typical geometries differ considerably from HOA [17]. WFS exhibits no pronounced sweet spot, the sampling artifacts are rather evenly distributed over the receiver area. The sampling artifacts may be perceived as coloration of the sound field [21].

The loudspeaker driving signals for WFS can be computed very efficiently by weighting and delaying the virtual source signals for the reproduction of virtual point sources and plane waves.

3.4 Numerical Methods

Besides HOA and WFS, several numerical methods of sound field reproduction [22,23,24] exist, the properties of which are typically somewhere between HOA and WFS. The advantage of these numerical methods are very flexible loudspeaker layouts. The drawback is the fact that they are numerically complex compared to the analytical solutions given by HOA and WFS, and that they do not provide such a high degree of flexibility.

3.5 Generalized Panning Techniques

Currently the most widely used method for creating a spatial sound impression is still based on the stereophonic approach described in Sect. 2. Panning techniques

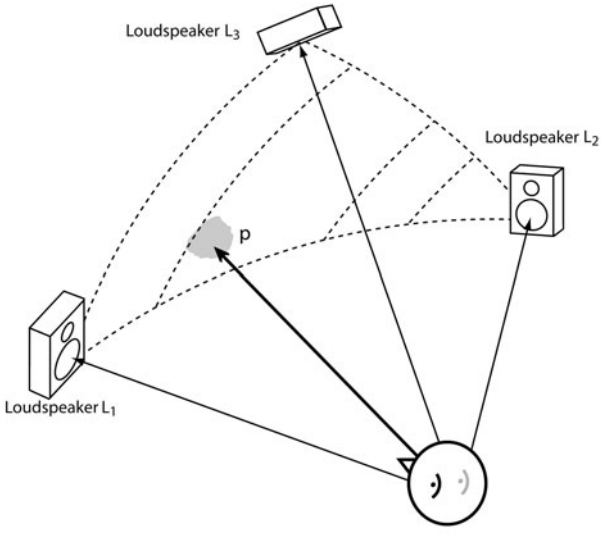


Fig. 4. Three-dimensional amplitude panning with amplitude factors calculated by projecting the target direction on a vector base spanned by loudspeaker triplets L_1, L_2, L_3

exploit the amplitude and delay differences between a low number of loudspeaker signals to create the illusion of a phantom source. For reproduction in a two-dimensional plane (e.g. 5.1 systems) normally a pair, for three-dimensional setups a triple or quadruple of loudspeakers is used. *Vector Base Amplitude Panning* (VBAP) [25] can be regarded as a generalization of amplitude panning. In the three-dimensional case, phantom sources can be placed on triangles spanned by the closest three loudspeakers (see Fig. 4).

Panning techniques allow flexible loudspeaker layouts but have a pronounced sweet spot. Outside this sweet spot, the spatial impression is heavily distorted and also some impairment in terms of sound color may occur.

4 Object-Based Rendering of Audio Scenes

Depending on how the loudspeaker driving signals are computed, two rendering techniques can be differentiated for most of the introduced audio reproduction approaches. On the one hand this is *data-based rendering* and on the other hand *model-based rendering*. In data-based rendering the entire audio scene is captured by a suitable number of microphones. Post-processing (encoding) is applied to the microphone signals if required for the particular technique under consideration. Typically circular or spherical microphone arrays are used for these approaches. The encoded signals are then stored or transmitted and the loudspeaker driving signals are computed from them. The benefit of this approach is that arbitrarily complex sound fields can be captured, the drawbacks are the technical complexity, the required storage and transmission capacity and the limited flexibility in post-processing.

Model-based rendering is based on the concept of using a parameterizable spatio-temporal model of a virtual source. Point sources and plane waves are the most frequent models used here, however, more complex models and superpositions of these simple sources are also possible. The signal of the virtual source together with its parameters is stored or transmitted. The receiver takes care that the desired audio scene is rendered appropriately for a given reproduction system. The benefits of this approach are the flexibility with respect to the reproduction system, the possibility for post-processing of the audio scene by modifying the parameters of the virtual source and the lower requirements for storage capacities for rather simple audio scenes. A drawback is that ambient sounds, like applause, are hard to model. Model-based rendering is termed as object-based audio since the sources in the scene are regarded as objects.

In practice, a combination of both approaches is used. The encoded microphone signals can be interpreted as a special source in model-based rendering and hence can be included in the scene description. Most of the discussed systems in Section 3 support both data-based and model-based rendering and also hybrid methods.

5 Formats

Along with the trend from channel-based to object-based audio reproduction there comes the necessity to store audio and metadata. Instead of storing a collection of audio tracks representing the loudspeaker signals, a so-called *audio scene* is created. Such an audio scene consists of sounding objects which are associated with input (source) signals, a certain position or trajectory in the virtual space and other source parameters. In addition to source objects there can also be objects describing the acoustical properties of a virtual room.

A wide variety of object-based audio reproduction systems is already in existence, both academic prototypes and commercially available implementations. Most of them use non-standard storage formats which are tailored to a single setup and in many cases contain implementation-specific data. Some systems are based on a specific digital audio workstation software – often using custom-made plugins – and use its native storage format including track envelope data for the dynamic control of source parameters. Although this may work very well on one single system, it is in most cases impossible to share audio scenes between different reproduction systems without major customization on the target system. It is crucial for a audio reproduction system to have content available, therefore it is desirable to establish a common storage format for audio scenes to be exchanged between different venues. Even if some fine-tuning is still necessary to adapt a scene to the acoustical conditions on location, this would very much facilitate the process.

There are two different paradigms for storing audio scenes, (1) to create a single file or stream which contains both audio data and scene data, or (2) to create one file for the description of the scene which contains links to audio data stored in separate audio files or streams. An advantage of the former method is its compactness and the possibility to transmit all data in one single stream. The latter method allows more flexibility as audio data can be edited separately or can be completely exchanged and several versions of an audio scene can be created with the same audio data. Another important aspect of a scene format is if it is stored in a binary file or in a text file. Binary files are typically smaller and their processing and transmission is more efficient. They are also the only feasible option if audio data and scene data are combined into one file. Text files have the advantage that they can be opened, inspected and edited easily with a simple text editor. Text-based formats can normally be extended more easily than binary formats. There are several markup languages which can be used to store data in a text file in a structured manner. One of the most widespread is the *eXtensible Markup Language* (XML). Many tools and software libraries to read, manipulate and store XML data are available.

Most of todays high resolution spatial audio reproduction systems have a means of controlling the audio scene parameters in realtime. This can be done by sending messages to the reproduction software, for example via network sockets. These messages can be collected, tagged with timestamps and written to a file. This way the realtime control format is also used as a storage format.

This paradigm is for example used in the *Spatial Sound Description Interchange Format* (SpatDIF) [26]. Because it is just an unstructured stream of messages, it is hard to make meaningful changes later.

In the following sections a few standardized formats are presented which could be suitable as an exchange format for spatial audio scenes. Thereafter, in Sect. 5.4, the *Audio Scene Description Format* (ASDF) is presented which is still in development and which tries to address the mentioned shortcomings of the other formats.

5.1 VRML/X3D

The *Virtual Reality Modeling Language* (VRML) is a format for three-dimensional computer graphics mainly developed for displaying and sharing of 3D models on the internet. Its scene description is based on a single scene graph, which is a hierarchical tree-like representation of all scene components. Geometrical objects are placed in local coordinate systems which can be translated/scaled/rotated and also grouped and placed in other coordinate systems and so on. Light sources, camera views and also audio objects have to be added to the same scene graph. To add an audio object to the scene graph, a `Sound` node has to be used. This node contains an `AudioClip` node which holds the information about the audio file or network stream to be presented. The format of the actual audio data is not specified by the standard. All elements of the scene graph can be animated with the so-called `ROUTE` element. This, however, is quite cumbersome for complex animations, therefore in most cases the built-in ECMAScript/JavaScript interpreter is used. To enable user interaction, mouse-events can be defined and can be bound to any visual element in the scene graph.

The use of a scene graph to represent a three-dimensional scene is very widespread in computer graphics applications. It is possible to combine very simple objects – mostly polygons – to more complex shapes and then combine those again and again to create high level objects. When transforming such a high level object, the transformation is automatically applied to all its components. In pure audio scenes, sounding objects normally consist of only one or a few parts and an entire scene often contains only a handful of sources. Using a scene graph in such a case would make the scene description overly complicated. The far worse disadvantage, however, is the distribution of the timing information. The timing of sound file playback is contained in the respective `Sound` node, the timing information of animations is spread over `ROUTE`s, interpolators and scripts. This makes it essentially impossible to edit the timing of a scene directly in the scene file with a text editor.

The VRML became an ISO standard in 1997 with its version 2.0, also known as VRML97. It has been superseded by *eXtensible 3D* (X3D) [27], which is an ISO standard since 2004. X3D consists of three different representations: the classic VRML syntax, a new XML syntax and a compressed binary format for efficient storage and transmission.

5.2 MPEG-4 Systems/AudioBIFS

The ISO standard MPEG-4 contains the *BInary Format for Scenes* (BIFS) which incorporates the VRML97 standard in its entirety and extends it with the ability to stream scene metadata together with audio data. The used audio codecs are also defined in the MPEG standard. The spatial audio capabilities – referred to as *(Advanced)AudioBIFS* [28] – were also extended by many new nodes and parameters.

Among the new features is the `AcousticMaterial` node, which defines acoustical properties like reflectivity (`reffunc`) and transmission (`transfunc`) of surfaces, the `AudioFX` node to specify filter effects in the *Structured Audio Orchestra Language* (SAOL) and the ability to specify virtual acoustics in both a physical and a perceptual approach. For the latter, the `PerceptualParameters` node with parameters like `sourcePresence` and `envelopment` can be used. Another new feature is the `DirectiveSound` node, used to specify source directivity.

AudioBIFS is a binary format which is designed to be streamed over a network. As a tool for easier creation and editing of scenes there is also a text-based representation, the *Extensible MPEG-4 Textual Format* (XMT). It comes in two variants: XMT-A has a syntax very similar to X3D (see Sect. 5.1), XMT-Ω is modeled after SMIL (see Sect. 5.3). However, the XMT is not a presentation language on its own, it has always to be converted to the binary format before it can be transmitted or played back.

AudioBIFS as part of MPEG-4 Systems became an ISO standard in 1999, but has evolved since. In its most recent update – AudioBIFS v3 [29] – several features were added, among them the `WideSound` node for source models with given shapes and the `SurroundingSound` node with the `AudioChannelConfig` attribute which allows to include Ambisonics signals and binaural signals into the scene.

AudioBIFS would definitely have all the features necessary to store spatial audio scenes. However, because of the huge size and complexity of the standard, it is very hard to implement an en- and decoder. No complete library implementation of MPEG-4 Systems is available.

5.3 SMIL

Contrary to the aforementioned formats, the XML-based *Synchronized Multimedia Integration Language* (SMIL, pronounced like "smile") is not able to represent three-dimensional content. Its purpose is the temporal control and synchronization of audio, video, images and text elements and their arrangement on a 2D screen. The SMIL is a recommendation of the *World Wide Web Consortium* since 1998, the current version (SMIL 3.0) was released in 2008 [30].

All SMIL functionality is organized in modules, for example *MediaDescription*, *PrefetchControl* and *SplineAnimation*. Different sets of modules are combined to language profiles tailored for different applications and platforms. With the *3GPP SMIL Language Profile* the SMIL is used for *Multimedia Messaging Service* (MMS) on mobile phones.

The central part of a SMIL document is a timeline where media objects can be placed either relative to other objects or by specifying absolute time values. The timing does not have to be static, interactive presentations can be created where the user dictates the course of events e.g. by mouse clicks. Animations along 2D-paths are possible with the `animateMotion` element. The temporal structure is mainly defined by `<seq>`-containers ("sequence"), whose content elements are played consecutively one at a time, and by `<par>`-containers ("parallel"), whose content elements start all at the same time. Of course, these containers can be arbitrarily nested giving possibilities ranging from simple slide shows to very complex interactive mega-presentations. Inside of the time containers, media files are linked to the SMIL file with `<img>`, `<audio>`, `<text>` and similar elements.

SMIL has very limited audio capabilities. Except for the temporal placement, the only controllable parameter of audio objects is the sound level, given as a percentage of the original volume. The SMIL format itself is especially not able to represent 3D audio scenes, but it can either be used as an extension to another XML-based format or it can be extended itself. To extend another XML-based format with SMIL timing features, the W3C recommendation *SMIL Animation* [31] can be utilized. This was done, for example, in the wide-spread *Scalable Vector Graphics* (SVG) format. However, *SMIL Animation* is quite limited because a "flat" timing model without the powerful time containers (like `<par>` and `<seq>`) is used. A more promising approach would be to extend the SMIL with 3D audio features. An example for such an extension is given in [32], where the SMIL was extended with the so-called *Advanced Audio Markup Language* (AAML).

5.4 ASDF

The *Audio Scene Description Format* (ASDF) [33] is an XML-based format which – contrary to the aforementioned formats – has a focus on pure audio scenes. It is still in an early development state, but basic functionality is already available in the *SoundScape Renderer* (SSR) [34].

The ASDF aims at being both an authoring and a storage format at the same time. In absence of a dedicated editing application, scene authors should still be able to create and edit audio scenes with their favorite text editor. The ASDF does not try to cover every single imaginable use case (like MPEG-4 does), but just follows the development of current audio reproduction techniques (see Sect. 3) and intends to provide a lowest-common-denominator description of scenes to be rendered by these techniques. To ensure the smooth exchange of scenes between different systems, it is independent of the rendering algorithm and contains no implementation-specific or platform-specific data. Requirements and implementation issues are discussed within the scientific community and with partners from the industry. The goal is to collaboratively develop an open format which is easy to implement even with limited resources. A reference implementation will be provided in form of a software library.

Although the ASDF is capable of representing three-dimensional audio scenes, there is also a simplified syntax available to describe two-dimensional scenes in a

horizontal plane. The ASDF does not have a hierarchical scene graph, it is rather using SMIL's timeline and time container concept (see Sect. 5.3). In its easiest form, the ASDF is used to represent static scenes, where source positions and other parameters do not change. If source movement is desired, trajectories can be assigned to sources or groups of sources.

In its current draft proposal, the ASDF is a stand-alone format, but it is planned to become an extension to SMIL. This way, a SMIL library can be used for media management and synchronisation and only the spatialization aspects have to be newly implemented. As a positive side effect of using SMIL, videos, images and texts can be easily synchronized with the audio scene and displayed on a screen.

When extending SMIL, it is important to separate the 3D audio description from SMIL's 2D display layout. Adding 3D audio features is not just a matter of adding a third dimension to the available 2D elements like `<layout>` and `<region>` (as done in [32]), because 2D has a different meaning in screen presentations and in spatial audio presentations. In the former case, the most natural choice for a 2D plane is of course the plane where the screen is part of. In the latter case, however, it makes much more sense to choose the horizontal plane as a 2D layout. If reproduction systems for spatial audio are limited to two dimensions, it will be in the most – if not all – cases the horizontal plane.

6 Applications

Most of the existing technical solutions for high resolution spatial audio reproduction are limited to one specific reproduction method. The required number and geometrical setup of loudspeakers differ considerably for different reproduction methods on the one hand, the digital signal processing for creation of the loudspeaker signals on the other.

In order to investigate the potential of object-based audio, the *SoundScape Renderer* (SSR) [34] has been implemented. The SSR supports a wide range of reproduction methods including binaural reproduction (Sect. 3.1), HOA (Sect. 3.2), WFS (Sect. 3.3) and VBAP (Sect. 3.5). It uses the ASDF as system independent scene description. The implementation allows for a direct comparison of different sound reproduction methods.

As one common interface is provided for different rendering backends, *system-independent mixing* can be performed, which means that a spatial audio scene can be created in one location (e.g. a studio with an 8-channel VBAP system) and performed in another venue (e.g. a concert hall with a several-hundred-channel WFS system). Figure 5 shows two screenshots of the graphical user interface of the SSR where the same scene is rendered with two different reproduction setups. With even less hardware requirements, the scene could also be created using binaural rendering and a single pair of headphones [35]. In any case, final adjustments may be necessary to adapt to the reproduction system and the room acoustics of the target venue.

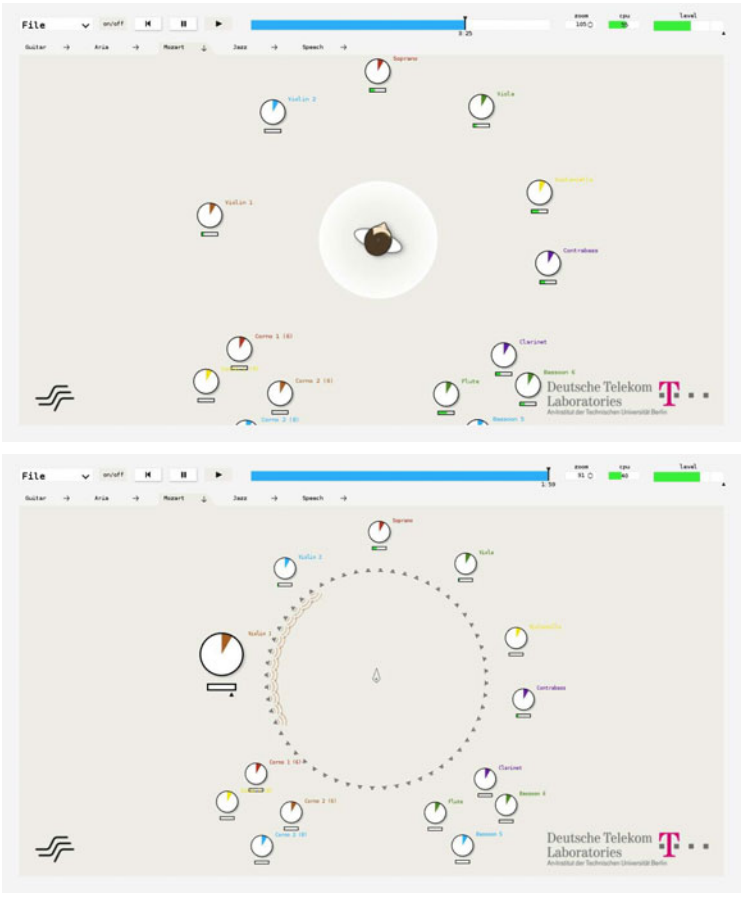


Fig. 5. The graphical user interface of the *SoundScape Renderer* reproducing the same scene with binaural rendering (top) and WFS (bottom)

7 Conclusions

Traditionally, techniques for capture, transmission and reproduction of spatial audio have been centered around a channel-based paradigm. Technically and practically this was feasible due to the low number of required channels and the restricted geometrical layout of the reproduction system. In practice, typical audio scenes contain more audio objects than reproduction channels which makes the channel-based approach also quite efficient in terms of coding and transmission.

Advanced loudspeaker based audio reproduction systems use a high number of channels for high-resolution spatial audio reproduction. It is almost impossible to predefine a geometrical setup for widespread deployment of such techniques. Here object-based audio scene representations are required that decouple the

scene to be reproduced from the geometrical setup of the loudspeakers. The rendering engine at the local terminal has to generate suitable loudspeaker signals from the source signals and the transmitted scene description.

Object-based audio also offers a number of benefits. Two major ones are: (1) efficient manipulation of a given audio scene, and (2) independence of the scene description from the actual reproduction system. In traditional channel-based audio, all audio objects that compose the scene are down-mixed to the reproduction channels. This makes it almost impossible to modify the scene when only having access to the down-mixed channels. The object-based approach makes such modifications much easier.

An variety of reproduction systems are currently being used or proposed for the future. Most likely none of these systems will be the sole winner in the future. Currently most audio content is either produced only for one reproduction system or has been produced separately for more than one. This situation leads to a number of proposals for the automatic up- and down-mixing of stereophonic signals. An object-based mixing approach will provide a number of improvements since it allows a very flexible rendering of audio scenes with respect to a given reproduction system.

A number of proposed formats for the object-based representation of audio(-visual) scenes exist. One of the most powerful formats, MPEG-4, is technically very demanding. This has lead to a number of other formats that are specialized for a specific application area. The ASDF focuses explicitly on the easy exchange of audio scenes.

In order to illustrate and investigate the benefits of object-based audio, the *SoundScape Renderer* has been developed. It fully separates the scene description from the audio reproduction approach. The SSR generates the loudspeakers signals for a variety of audio reproduction approaches in real-time from the scene description. Furthermore, real-time interaction with the audio scene is possible. Traditional channel-based approaches can not provide this degree of flexibility.

References

1. Leakey, D.: Further thoughts on stereophonic sound systems. Wireless World 66, 154–160 (1960)
2. Williams, M.: Unified theory of microphone systems for stereophonic sound recording. In: 82nd Convention of the Audio Engineering Society (March 1987)
3. Wendt, K.: Das Richtungshören bei Zweikanal-Stereophonie. Rundfunktechnische Mitteilungen 8(3), 171–179 (1964)
4. Theile, G.: Zur Theorie der optimalen Wiedergabe von stereophonen Signalen über Lautsprecher und Kopfhörer. Rundfunktechnische Mitteilungen 25, 155–169 (1981)
5. Gernemann-Paulsen, A., Neubarth, K., Schmidt, L., Seifert, U.: Zu den Stufen im "Assoziationsmodell". In: 24. Tonmeistertagung (2007)
6. Blumlein, A.: Improvements in and relating to sound-transmission, sound-recording and sound-reproducing systems. British Patent Specification 394325 (1931)
7. Thiele, H.H.K. (ed.): 50 Jahre Stereo-Magnetbandtechnik. Die Entwicklung der Audio Technologie in Berlin und den USA von den Anfängen bis 1943. Audio Engineering Society (1993)

8. Woodward, J.: Quadraphony–A Review. Journal of the Audio Engineering Society 25(10/11), 843–854 (1977)
9. Theile, G., Plenge, G.: Localization of lateral phantom sources. Journal of the Audio Engineering Society 25, 196–200 (1977)
10. Blauert, J.: Spatial Hearing: The Psychophysics of Human Sound Localization. MIT Press, Cambridge (1996)
11. Lindau, A., Hohn, T., Weinzierl, S.: Binaural resynthesis for comparative studies of acoustical environments. In: 122nd Convention of the Audio Engineering Society (May 2007)
12. Møller, H.: Reproduction of artificial-head recordings through loudspeakers. Journal of the Audio Engineering Society 37, 30–33 (1989)
13. Daniel, J.: Représentation de champs acoustiques, application à la transmission et à la reproduction de scènes sonores complexes dans un contexte multimédia. PhD thesis, Université Paris 6 (2000)
14. Poletti, M.: Three-dimensional surround sound systems based on spherical harmonics. Journal of the Audio Engineering Society 53(11), 1004–1025 (2005)
15. Ahrens, J., Spors, S.: An analytical approach to sound field reproduction using circular and spherical loudspeaker distributions. Acta Acoustica united with Acoustica 94(6), 988–999 (2008)
16. Fazi, F., Nelson, P., Christensen, J., Seo, J.: Surround system based on three dimensional sound field reconstruction. In: 125th Convention of the Audio Engineering Society (2008)
17. Spors, S., Ahrens, J.: A comparison of Wave Field Synthesis and Higher-Order Ambisonics with respect to physical properties and spatial sampling. In: 125th Convention of the Audio Engineering Society (October 2008)
18. Ahrens, J., Spors, S.: Alterations of the temporal spectrum in high-resolution sound field reproduction of different spatial bandwidths. In: 126th Convention of the Audio Engineering Society (May 2009)
19. Berkhout, A.: A holographic approach to acoustic control. Journal of the Audio Engineering Society 36, 977–995 (1988)
20. Spors, S., Rabenstein, R., Ahrens, J.: The theory of Wave Field Synthesis revisited. In: 124th Convention of the Audio Engineering Society (May 2008)
21. Wittek, H.: Perceptual differences between Wavefield Synthesis and Stereophony. PhD thesis, University of Surrey (2007)
22. Kirkeby, O., Nelson, P.: Reproduction of plane wave sound fields. Journal of the Acoustic Society of America 94(5), 2992–3000 (1993)
23. Ward, D., Abhayapala, T.: Reproduction of a plane-wave sound field using an array of loudspeakers. IEEE Transactions on Speech and Audio Processing 9(6), 697–707 (2001)
24. Hannemann, J., Leedy, C., Donohue, K., Spors, S., Raake, A.: A comparative study of perceptual quality between Wavefield Synthesis and multipole-matched rendering for spatial audio. In: IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) (April 2008)
25. Pulkki, V.: Virtual sound source positioning using Vector Base Amplitude Panning. Journal of the Audio Engineering Society 45(6), 456–466 (1997)
26. Peters, N.: Proposing SpatDIF – the spatial sound description interchange format. In: International Computer Music Conference (August 2008)
27. Web3D Consortium: eXtensible 3D, X3D (2004), http://www.web3d.org/x3d/
28. Väänänen, R., Huopaniemi, J.: Advanced AudioBIFS: Virtual acoustics modeling in MPEG-4 scene description. IEEE Transactions on Multimedia 6(5), 661–675 (2004)

29. Schmidt, J., Schröder, E.F.: New and advanced features for audio presentation in the MPEG-4 standard. In: 116th Convention of the Audio Engineering Society (May 2004)
30. World Wide Web Consortium: Synchronized Multimedia Integration Language, SMIL 3.0 (2008), http://www.w3.org/TR/SMIL3/
31. World Wide Web Consortium: SMIL Animation (2001), http://www.w3.org/TR/smil-animation/
32. Pihkala, K., Lokki, T.: Extending SMIL with 3D audio. In: International Conference on Auditory Display (July 2003)
33. Geier, M., Spors, S.: ASDF: Audio Scene Description Format. In: International Computer Music Conference (August 2008)
34. Geier, M., Ahrens, J., Spors, S.: The SoundScape Renderer: A unified spatial audio reproduction framework for arbitrary rendering methods. In: 124th Convention of the Audio Engineering Society (May 2008)
35. Geier, M., Ahrens, J., Spors, S.: Binaural monitoring of massive multichannel sound reproduction systems using model-based rendering. In: NAG/DAGA International Conference on Acoustics (March 2009)

Using Thematic Ontologies for User- and Group-Based Adaptive Personalization in Web Searching

Alexandros Paramythis[1], Florian König[1], Christian Schwendtner[1], and Lex van Velsen[2]

[1] Johannes Kepler University
Institute for Information Processing and Microprocessor Technology (FIM)
Altenbergerstraße 69, A-4040 Linz, Austria
{alpar,koenig,schwendtner}@fim.uni-linz.ac.at
[2] University of Twente
Department of Technical and Professional Communication
P.O. Box 217, 7500 AE Enschede, The Netherlands
l.s.vanvelsen@utwente.nl

Abstract. This paper presents Prospector, an adaptive meta-search layer, which performs personalized re-ordering of search results. Prospector combines elements from two approaches to adaptive search support: (a) collaborative web searching; and, (b) personalized searching using semantic metadata. The paper focuses on the way semantic metadata and the users' search behavior are utilized for user- and group- modeling, as well as on how these models are used to re-rank results returned for individual queries. The paper also outlines past evaluation activities related to Prospector, and discusses potential applications of the approach for the adaptive retrieval of multimedia documents.

1 Introduction

The phenomenal growth of the web in the past decade has resulted in an unprecedented amount of information being available in accessible electronic form, and this trend can only be expected to strengthen in coming years. This proliferation of information, though, has rendered locating the items of information that are indeed interesting to a user an increasingly difficult task. To address the needs of modern web searching, several approaches have been proposed and practically applied that improve upon traditional term-matching information retrieval techniques.

Significant innovations include the works reported in [1] and [5] that have exploited document connectivity information to significantly improve retrieval quality. More recently, other researchers have sought to exploit context as a means of supplementing vague queries and so "guiding" search [6]. A different line of work has looked at clustering techniques as a way to impose order on a collection of search results, with a view to identifying different conceptual groupings of results [3][4].

In the realm of collaborative search systems, which utilize the collective experiences of like-minded groups of users to improve upon search results, a representative and widely acclaimed system is I-SPY [8][9]. I-SPY implements an adaptive collaborative search technique that enables it to selectively re-rank search results according

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 18–27, 2010.

to the learned preferences of a community of users. Effectively I-SPY actively promotes results that have been previously favored by community members during related searches so that the most relevant results are at the top of the result list [8]. I-SPY monitors user selections or "hits" for a query and builds a model of query-page relevance based on the probability that a given page will be selected by the user when returned as a result to a specific query.

Along a different line of work, researchers have addressed the use of semantic metadata to represent user interest profiles and adapt searches so that results better fit those profiles. Two such systems based on metadata provided by the Open Directory Project[1] are the ones described in [10], and [2]. The first paper describes Persona, a system which utilizes ODP metadata for creating taxonomies of user interests and disinterests and tree coloring to represent user profiles. Taxonomy nodes visited are 'colored' by the number of times they have been visited, by user ratings if available, and by the URLs associated with the node [10]. In the second paper, Chirita et al. [2] have used ODP metadata to create user profiles, and then used various approaches to calculating the distance between a given search result item and the user's profile, to decide that item's rank. Users pre-select ODP categories that they are interested in for the creation of their profiles; the system does not have an adaptive component, so these profiles do not evolve over time. The distance calculation approaches range from ones based primarily on graph node distances, to a version of the PageRank [1] algorithm modified to include a measure of the semantic similarity between nodes in a taxonomy. User-based experiments have shown that these approaches to search personalization deliver superior results to their non-personalized counterparts [2].

This paper presents the Prospector system [7], an adaptive meta-search layer on top of mainstream search engines. Prospector implements a hybrid approach, whereby: (a) ODP ontological metadata is used as the basis for dynamically maintained models that capture the search preferences of individuals; and, (b) group models are created and maintained alongside the individual user models, and subsequently used to improve upon search results of individuals belonging to a group.

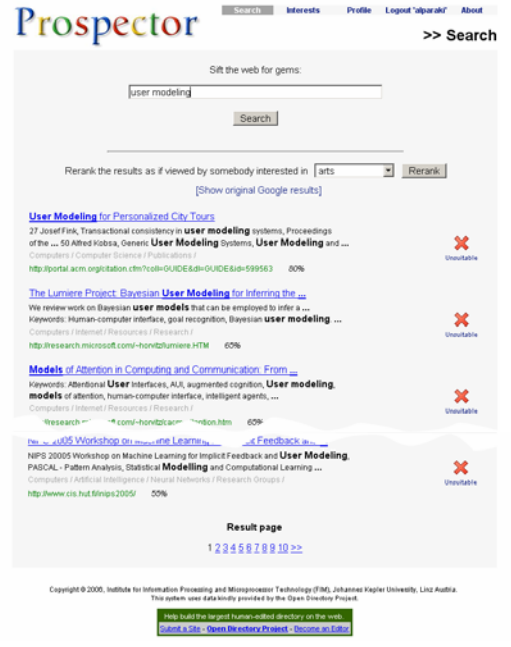
2 The Prospector System

Prospector is a generic front-end to mainstream search engines. There have been two versions of the system so far, the first described in [7] and the second in this paper. The most important difference between the two versions is the employment of custom algorithms for the modeling and adaptation processes, which are described herein (the previous version used item weighting algorithms). This change has brought about: (a) improved system effectiveness in adaptively reordering search results for individuals and thematic groups; and (b) increased comprehensibility by end users of result item relevance ratings, and of their own models / profiles.

Here, as well as in [7], we present the system running on top of the Google search engine. Due to space restrictions, we only provide a brief overview of system interactivity here, focusing on features that have changed since the first version. A more complete account of the interactive aspects of the system can be found in [7].

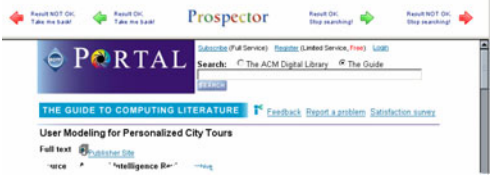
[1] Open Directory Project: http://dmoz.org/

(a) Setting up a user profile



(b) Searching

(d) Inspecting and manipulating the user model



(c) Rating search result items

Fig. 1. A quick visual tour of Prospector

In Prospector, a basic anonymous search returns exactly the same results as a search made directly on the Google site. Adaptivity comes into play in two guises: firstly, while still remaining anonymous, the user can have the results re-ranked by the system, according to thematically-based group models; and, secondly, users can register (and log in), progressively building up their personal interest profile, which is then used to automatically re-rank search results.

Modeling of search behavior in Prospector takes place on the basis of ODP metadata. Group modeling is based on a fixed set of groups, one for each of the top-level categories in ODP. In compensation to the absence of dynamically determined groups, the system allows users to define the “degree of affiliation” to these groups, through the definition of their level of interest in the respective thematic categories (see Fig.1-a). After the creation of the user profile, which in turn results in the creation of a corresponding user model, the user can perform queries and have the results

re-ranked according to their individual user model, as well as according to the models of groups that the user is interested in (see Fig.1-b). Users are able to rate individual search result items, either from within the results' page (Fig.1-b), or using the controls in the "rating frame" which appears when they follow the link of a search result item (Fig.1-c). Finally, Prospector allows users to both inspect and manipulate their personal model, to better and faster fine-tune it to their search preferences (Fig.1-d).

As already mentioned, Prospector is an effort to create a hybrid web search support layer, which uses concepts and techniques both from collaborative web searching, and from semantically enriched result filtering / re-ordering, to achieve high levels of personalization of search results. The rest of the paper will provide a detailed view of the modeling and adaptation algorithms used in the system, as well as on the preliminary conclusions we have drawn from our work on the system thus far.

3 Ontology-Based Modeling of Search Behavior

3.1 Basic Concepts

Before we proceed to presenting the modeling and re-ranking algorithms used in Prospector, we need to establish the basic concepts used.

To start with, both user and group models in Prospector are, in effect, overlay models over the ODP category ontology[2]. Specifically, the models are structured hierarchically following the topic relations in the ODP ontology, and each node in a user- or group- model contains the likelihood that the user or group, respectively, are interested in items (web sites) that are associated with the node's category. A category "path" is the branch in the hierarchy that has a specific category as its end node.

Based on the above, the following conventions are used in the rest of this paper: U denotes the set of all system users. G denotes the set of top-level ODP categories used by Prospector as groups. When a user first registers in the system, they are presented with a form in which they specify their level of interest in each of these categories / groups using a 5-point scale. This value is denoted as $Interest(u, g)$, $u \in U$ (the current user), and $g \in G$; the value space for $Interest(u, g)$ is therefore $\{0, 1, 2, 3, 4, 5\}$, with a value of zero signifying no interest in the respective category / group.

Using the above, we define $AverageInterest(g)$ to be:

$$AverageInterest(g) = \frac{\sum_{k=1}^{N} Interest(u_k, g)}{N} \tag{1}$$

where N is the number of users for whom $Interest(u, g) > 0$. In other words, $AverageInterest(g)$, is the average interest value for g, over all users with any interest in g.

This, in turn, is used to determine the influence of group models on the individual user model, and vice versa. This value is encapsulated in an influence coefficient:

$$Influence(u, g) = \frac{Interest(u, g)}{AverageInterest(g)} \frac{1}{N} \tag{2}$$

[2] In the ODP dataset, categories are termed "topics".

where, again, N is the number of users for whom $Interest(u, g) > 0$. Note that for a given user, this coefficient is defined for all groups in which the user has expressed interest in, and is quite likely to differ for each such group.

We use R to denote the set of search result items returned by a query q, and $c(r)$ (or, simply, c) to denote the primary ODP category associated with a search result item r, with $r \in R$. This category is derived as follows. Let $C(r)$ be the set of all ODP categories associated with result item r. For each category $c \in C(r)$, $Items(c)$ denotes the total numbers of items (links, PDF documents, RSS feeds, etc.) associated with that category. The strategy used in the current version of Prospector is to use the category with the highest number of items associated with it, i.e.,

$$c(r) = c \in C(r) \quad and \quad Items(c) \geq Items(c'), \forall c' \in C(r) \tag{3}$$

Finally, for an ODP category c, we define $Depth(c)$ to be the number of path elements that the category comprises (e.g., $Depth$("Top/Computers/Software") = 3).

3.2 User Modeling Using Search Result Item Ratings

As already seen, users have the possibility to rate a specific search result r. Such ratings modify the likelihood that the user is interested in the respective category $c(r)$ – denoted as c for brevity. The desiderata for likelihood modifications in this context are: the first few ratings for a category should have "immediate" effects, making the system quickly converge to the user's preferences with respect to the category at hand; for categories with well-established user preferences (e.g., categories with many negative ratings), an "opposite" rating, i.e., a rating that goes against the usual user ratings for the category, should not have major effects on the likelihood value.

To attain the desired behavior, the likelihood $P(u, c)$ that user u is interested in items associated with the ODP category c is defined as follows:

$$P(u,c) = \frac{\cos\left(\left(1 - \frac{positive(u,c)+5}{10}\right)\cdot\pi\right)}{2} + 0.5 \tag{4}$$

Where, $positive(u, c)$ is defined as the sum of the number of positive ratings made by user u for items in category c, minus the sum of the number of negative ratings made by user u for items in category c. The value of $positive(u, c)$ is constrained to the space [-5 .. 5]. The actual amount by which the likelihood value in the user model changes is then: $\Delta P(u,c,a) = P_{after}(u,c) - P_{before}(u,c)$, where P_{before} and P_{after} refer to $P(u, c)$, after and before the user rating a respectively.

When a rating is made, not only the likelihood value for the specific search result item's category is modified, but also the values of all categories that are its ancestors. In other words, ratings for items in a category affect the entire category "path". The effects of a rating are scaled to reflect the distance of path nodes from the actual category for which the rating was made – closer nodes are affected more than distant ones. Thus, for all categories c' that are ancestors of c, and for a rating a we have:

$$\Delta P(u,c',a) = \left(P_{after}(u,c') - P_{before}(u,c')\right)\cdot\frac{Depth(c')}{Depth(c)} \tag{5}$$

To apply this propagating modification approach, it is of course necessary that the user model contains the entire path that corresponds to a category. If the path does not exist prior to a rating, it is created and populated on the fly using "bootstrap" likelihood values derived from the models of groups that the user is interested in, and from the user's own model, if the later contains any segment of the path. The actual topic rated on is set to 0.5 so as not to "surprise" users with large / low inherited values.

The approach to deriving likelihood values in this context is very similar to the one used for predicting values for unrated categories, and is discussed in section 4 below.

3.3 Group Modeling Using Search Result Item Ratings

When a rating is made, the group models for all groups / categories that the user is interested in are also modified. In essence, user ratings have similar effects on group models as they do for user models, but these effects are scaled to reflect both the user's own interest in a group, and the number of users that are interested in that group. The rest of this section will provide formal definitions for the above.

To start with, we define $P(g, c)$ to be the likelihood that a user that is highly interested in group g (i.e., has an interest value of 5 for g) will be interested in items associated with the ODP category c. When the system starts for the first time, all group models are initialized to reflect a high interest in all categories that are subcategories of that group (i.e., $P(g, c) = 1.0$, when $c=g$).

When a rating is made for a category c, by user u, then for all groups g that u is interested in, $P(g, c')$ (where c' is an ancestor of c, including c itself) is modified as follows:

$$\Delta P(g,c',a) = \Delta P(u_g,c') \cdot \frac{Depth(c')}{Depth(c)} \cdot Influence(u,g) \tag{6}$$

where, $\Delta P(u_g,c',a)$ denotes the modification in $P(g, c')$ that would have resulted if an imaginary user u_g representing the entire group had made the rating a. $\Delta P(u_g,c',a)$ is calculated using a formula similar to the one in equation (4), with $positive(u, c)$ replaced with $rating(u_g, c, a)$, defined as follows:

$$rating(u_g,c,a) = 1 - \frac{\arccos\left(\left(P(g,c) - 0.5\right)\cdot 2\right)}{\pi} + increment(a) \tag{7}$$

with $increment(a)$ given the value of 0.1 for positive ratings and -0.1 for negative ratings. Note that as equation (6) suggest, the propagating modification approach is used for group models in the same way as for individual user models.

4 Ontology-Based Reordering of Search Results

The re-ranking of results is based on the likelihood that a given result will be of interest to a given user. The process of determining this is as follows: When a user performs a search, result items r that correspond to sites that exist in the dataset are associated with a category $c(r)$, as described in section 3.1 above. $P(u, r)$ denotes the likelihood that user u is interested in item r. This is defined as follows:

- If the item r cannot be associated with an ODP category, then $P(u, r) = 0.5$
- If the user's individual model already includes a value for $c(r)$ (on the basis of previous user ratings), that value is used verbatim
- If the user's individual model does not include a value for $c(r)$, then derive a value from the models of groups in which the user is interested, as well as from the user's own model, if the later contains values for ancestor categories of c

In the context of the third of the above cases, the following definitions are made: $P_{group}(u, c)$ denotes the likelihood that user u is interested in category c, as derived from the models of groups in which the user is interested; $P_{inherited}(u, c)$ denotes the likelihood that user u is interested in category c, as calculated from the user's own model, using ancestor categories of c. $P_{predicted}(u, c)$, which denotes the overall predicted likelihood that user u is interested in category c, is then defined as:

$$P_{predicted}(u,c) = \lambda \cdot P_{groups}(u,c) + (1-\lambda) \cdot P_{inherited}(u,c) \tag{8}$$

The factor $\lambda \in [0\ldots1]$ can be modified to favor predictions based on group models or predictions based on the user's own model. In the current version of Prospector, its value has been set to 0.75 (thus favoring groups), but its effect has not been experimentally validated yet. If either of $P_{group}(u, c)$ or $P_{inherited}(u, c)$ are not available, then the other one is used exclusively. If neither of them are available, a "default" likelihood of 0.5 is used instead. In the above equation, $P_{group}(u, c)$ is defined as:

$$P_{groups}(u,c) = \frac{\sum_{i=1}^{N}\left(\left(P(g_i,c)-0{,}5\right)\cdot\frac{Interest(u,g)}{5}+0.5\right)}{N} \tag{9}$$

where g_i are all the groups in which user u is interested. $P_{inherited}(u, c)$ is defined as:

$$P_{inherited}(u,c) = \frac{\sum_{i=1}^{N}\left(\left(P(u,c_i)-0.5\right)\cdot\frac{Depth(c_i)}{Depth(c)}+0.5\right)}{N} \tag{10}$$

where c_i are all the ancestor categories of c for which there already exists a node in the model.

Note that $P_{predicted}(u, c)$ is also used when "bootstrapping" a user model node, as described in section 3.2 above. When bootstrapping occurs within group models, an appropriately modified version of $P_{inherited}(u, c)$ is used instead.

5 Evaluation Activities

The version of the Prospector system described in this paper underwent formative user-centered evaluation at the University of Twente. Three different evaluation methods were applied in this study: thinking aloud, interviews and questionnaires. Data logs of the participants' activity in the system (including searching, viewing and rating results, inspecting and modifying their model, etc.) were also collected. A full

report of evaluation activities and obtained results is being prepared for separate publication. Below we summarize the most important, preliminary, findings.

During the evaluation, 32 participants were given the task of searching for youth hostels and museums of modern art in large European cities. These tasks have been chosen because the related information is relatively easy to find and is presented on many websites categorized by ODP. Thus, a user profile that ensures a high degree of personalization on these topics can be created in a small amount of time. For one city, the information had to be found with Google, for four cities with Prospector.

For the Google searches, and the last (fully personalized) Prospector searches, for a youth hostel and a museum of modern art, the participants had to rate the perceived relevance of the five highest ranked search results on a 7-point Likert scale, ranging from very irrelevant to very relevant. In the case of the youth hostel search the average Google relevance score ($M = 5.24$, $SD = .92$) did not differ from the average, fully personalized, Prospector relevance score ($M = 5.08$, $SD = .96$), $t(30) = .64$, n.s. In the case of the museum of modern art search, the Google relevance score, ($M = 3.69$, $SD = 1.24$) did differ from the fully personalized Prospector relevance score ($M = 2.78$, $SD = 1.09$), $t(31) = 3.93$, $p < .001$. Note that the latter scores are on the negative side of the scale, indicating that this appeared to be a difficult search for the participants.

The thinking-aloud protocols provided us with several causes for these results, the most important one being the effects of the ODP category 'news', which appeared to cause problems for the generation of appropriately personalized results. Many users had a high interest in news and scored it as such on their interest page. However, when they search for information that is not related to news (e.g., a youth hostel in Oslo) they are presented with search results focusing on (possibly outdated) news which interfere with their search goal. This happened particularly during the searches for museums of modern art. Although there is no easy solution to this problem, a possible approach would be to categorize news only as such on the basis of the age of the respective page (e.g., not older than 10 days); this would be more in line with users' perception of 'news'.

Other results derived from this study include the following:

- The current incarnation of the rating frame above each opened search result appears to feel "unnatural" to some users. Participants forgot to use it as they are used to working with the browser's back button, or did not use it correctly. For instance, they went back with the 'Result OK! Take me back' option and then removed the site with the 'Unsuitable' button. Consequently, the generation of the user model suffered from incorrect and / or missing usage data.
- The ODP categories, associated with a site, are sometimes wrong, which may influence the success of personalization. Sometimes, this is also affected by the exact URL used to refer to a site. For instance, the URL "en.wikipedia.org" is associated with 5 categories (i.e., "Arts / Television / Programs / Children's / Sesame Street"). None of these is indicative of the fact that Wikipedia is a free, on-line encyclopedia. In contrast, the URL "en.wikipedia.org/" (note the slash at the end of the URL), is associated with only one category: "Computers / Open Source / Open Content / Encyclopedias / Wikipedia". This issue resulted in users having a misrepresentation of (dis)interests in their user models, with ratings being assigned to wrong or irrelevant categories. This, in turn, sometimes resulted in unexpected results, with low relevance to the user's real interests.

– The wording of ODP categories brings along some serious problems when users indicate their interests or alter their user profile. Participants said they found the categories vague and needed more information about their meaning in order to indicate their interests or alter their profile properly. As a result, the success of personalization suffers: people may interpret categories wrong and create an incorrect user profile.

6 Conclusions and Discussion

This paper has presented the second version of the Prospector system, which uses thematic ontologies from the ODP project, and a set of custom algorithms for the adaptive reordering of search results on the basis of user- and group- models. Evaluation results have been very valuable in the iterative design process, leading to the currently under development third generation of Prospector, which addresses most of the issues mentioned in the previous section.

The proposed approach seeks to combine two sources of information for improving on the relevance of search results: (a) semantic information about the result items, and (b) individual- and group oriented- preferences and interests in thematic categories that directly relate to the aforementioned semantic information. Furthermore, it has been designed to be largely agnostic of the underlying search engine, and of item attributes (other than their thematic categories, and the relationships between those). These characteristics make this approach applicable "on top" of any user-oriented search system that maintains such semantic information for the items in its index, including multimedia search systems, such as, for instance, rich image collections with large ontologies and well-categorized items (e.g., www.istockphoto.com). The strengths of the approach lie with: the involvement of the user community in establishing the affinity of specific items to specific themes; and, the fact that the quality of system results increases with the number of individuals that use it, and with the number of ratings made by each individual.

The work we have done so far in applying this approach on open corpus document search with Prospector has provided a number of valuable lessons that should be heeded in any future work in this direction:

– The approach works best in contexts where disambiguation (due, e.g., to synonymity / homonymity issues) is required to provide relevant search results.
– The quality of the semantic information available for searchable items is of paramount importance. In this context, the correctness of existing semantic information is more central to the operation of the system, than the non-existence of said information. In other words, we have seen that the system can "cope" with missing pieces of categorization information better than with erroneous ones.
– The dependency on the availability of valid semantic information may render this approach inappropriate for employment in open corpora with little (if any) such information present. A better fit is envisaged for closed corpora, where semantic information is already available, or can be readily and progressively provided by the users of the search system. This would include systems where users can "tag" items in a structured or semi-structured manner.

References

1. Brin, S., Page, L.: The Anatomy of a Large-Scale Hypertextual Web Search Engine. Computer Networks and ISDN Systems 30(1-7), 107–117 (1998)
2. Chirita, P.A., Nejdl, W., Paiu, R., Kohlschütter, C.: Using ODP Metadata to Personalize Search. In: Proceedings of the 28th ACM International SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil. ACM, New York (2005)
3. Dell Zhang, Y.D.: Semantic, Hierarchical, Online Clustering of Web Search Results. In: Yu, J.X., Lin, X., Lu, H., Zhang, Y. (eds.) APWeb 2004. LNCS, vol. 3007, pp. 69–78. Springer, Heidelberg (2004)
4. Hamilton, N.: The mechanics of a deep net metasearch engine. In: Proceedings of the 12th International World Wide Web Conference, Budapest, Hungary (2003)
5. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. Journal of the ACM 46(5), 604–632 (1999)
6. Lawrence, S.: Context in Web Search. IEEE Data Engineering Bulletin 23(3), 25–32 (2000)
7. Schwendtner, C., König, F., Paramythis, A.: Prospector: An adaptive front-end to the Google search engine. In: Proceedings of the 14th Workshop on Adaptivity and User Modeling in Interactive Systems (ABIS 2006), held in the context of Lernen-Wissensentdeckung-Adaptivität 2006 (LWA 2006), October 9-11, pp. 56–61. University of Hildesheim, Hildesheim (2006)
8. Smyth, B., Balfe, E., Briggs, P., Coyle, M., Freyne, J.: Collaborative Web Search. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI 2003), Acapulco, Mexico, pp. 1417–1419. Morgan Kaufmann, San Francisco (2003)
9. Smyth, B., Freyne, J., Coyle, M., Briggs, P., Balfe, E.: I-SPY: Anonymous, Community-Based Personalization by Collaborative Web Search. In: Proceedings of the 23rd SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence, Cambridge, UK, pp. 367–380. Springer, Heidelberg (2003)
10. Tanudjaja, F., Mui, L.: Persona: A Contextualized and Personalized Web Search. In: Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS 2002), Hilton Waikoloa Village, Island of Hawaii, vol. 3, p. 67 (9). IEEE Computer Society, Los Alamitos (2002)

A Poset Based Approach for Condition Weighting

David Zellhöfer and Ingo Schmitt

BTU Cottbus
Department of Computer Science
Database and Information Systems Group
{david.zellhoefer,schmitt}@tu-cottbus.de

Abstract. Current research in multimedia retrieval (MR) does not satisfactorily mirror research results from psychology revealing a different significance of certain characteristics of a media object to a query in terms of similarity. Although the relevance of user-controlled condition weights has been demonstrated, there is a lack of systems supporting users in setting these weights.

In this work, we present a relevance feedback based approach that supports users to set condition weights in order to retrieve results from the MR system that are consistent with their perception of similarity. Condition weights are learned by a machine based learning algorithm from user preferences based on a partially ordered set.

1 Introduction

Traditional multimedia retrieval deals mainly with the extraction of features and their comparison on a machine level. This method disregards actual user needs. Common algorithms extract data from images, videos or audio in order to provide low-level semantic meta data to the retrieval system's algorithms. This low-level data cannot be interpreted directly by a human. In order to adjust search results to the user's expectations e.g. relevance feedback [1] is used, which relies on the user's willingness to evaluate a number of returned documents in terms of relevance[1] to improve the result quality during an iterative process.

Queries in a multimedia retrieval system combine different search paradigms such as information retrieval (IR) [2] and relational database (DB) queries [3,4]. These paradigms differ in the way in which document matches are expressed. While the first approach tries to model similarities using methods such as fuzzy set theory [5], the second relies on Boolean logic and crisp set theory. Particularly, the concept of similarity introduces both vagueness and subjectivity. Psychological experiments have shown that the perception of similarity is based on subjective criteria such as 'key stimuli' [6,7]. These results have been proven by IR research revealing that user-definable weights on query conditions contribute significantly to the user-satisfaction [8,9].

To combine different search paradigms in multimedia retrieval, a theoretical framework based on quantum logic [10] has been proposed [11]. It has also been shown that this approach is powerful enough to include weighted conditions into its logic [12].

[1] Usually only positive and negative examples are expressed.

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 28–39, 2010.

This paper is based on these results in order to provide a unified framework for multi-paradigm queries that has been postulated in recent works [13,14,15]. While there are other approaches combining different search paradigms in parallel connecting them afterwards, we can provide logic-based means to process both conjunctive and disjunctive combinations of conditions differing in their paradigm[2].

This paper focuses mainly on the formulation of condition weights for multimedia retrieval by the use of an iterative relevance feedback approach based on partially ordered sets (posets) for the following scenario. Based on a given logic-based query[3], initially unknown weights for all conditions have to be found in order to match the user's subjective preferences (see above). In our approach, the actual conditions of the query are fixed, but the condition weights are variable. The novelty of our approach is that we use machine learning to interpret the user's perception of similarity or relevance from a small amount of reviewed result objects in order to learn condition weights for a given logical query. In contrast to common approaches, such as Fagin and Wimmers' [16], these weights can be directly integrated into a query that does not leave the rules of a Boolean algebra.

In comparison to other machine based learning approaches, such as support vector machines (SVM), we can exploit the fact that we can use the user-defined logical query in addition to a user-provided poset in order to find condition weights reducing the issue to a non-linear optimization problem (see Sec. 4). Hence, we do not need a large training set of media objects for a neural network or a SVM to find actual characteristics of input media objects that form the user's perception of similarity. Consider the fact that the user will not be capable of providing a large amount of training objects such as images in general. Thus, we use the user-defined poset as input for our algorithm. Therefore, we can provide a more user-friendly system because the user interaction can be minimized by introducing an 'intelligent' algorithm in addition to a predictable evaluation behavior of the query itself that cannot be guaranteed by a SVM or the like.

This paper is organized as follows. Section 2 gives a description of our relevance feedback approach and focuses on the user interaction model and its interface metaphors. Additionally, requirements for a machine based learning algorithm are outlined. The following section discusses initial weighting schemes for the proposed process. Section 4 presents an algorithm for the derivation of condition weights from user-defined posets of result objects. Finally, we conclude in section 5.

2 User-Centered Relevance Feedback

The manual setting of user-controllable condition weights, particularly within complex multimedia queries, involving explicit conditions such as conditions on DB attributes or implicit low-level features[4], is a difficult task. Consider the following example: the user wants to find all visual media objects (photos and videos) that are similar to a given

[2] This holds for arbitrary recursive combinations of these as well.
[3] Using conjunctions, disjunctions and negation.
[4] Color and edge histograms, frequency analysis etc.

1. similarity to a given image
2. color = 'black and white'
3. category = 'socialist realism'
4. location ~ 'Berlin/Brandenburg'

Fig. 1. Example multimedia retrieval query

sample image[5], which are taken in black and white, belong to the art style category 'socialist realism' and are located in the geographical area of 'Berlin/Brandenburg' (see Fig. 1).

This query combines an implicit similarity query which is based on low-level features (1) with two database attribute comparisons (2,3) and a spatial proximity condition (4). Unfortunately, this common query-by-example (QBE) approach combined with keywords neither reveals the subjective relative importance of the conditions for the user, nor their logical connection (conjunction or disjunction).

With respect to research results in psychology [7] and information retrieval [8], it is import to take a subjective weighting of conditions into account. While it might be simple for the user to input certain weights on the given conditions in this example, the task becomes more difficult for complex queries. This is due to the user's inability to understand the semantics of all low-level features and their interactions amongst each other and with other conditions. Thus, the user can hardly be expected to input a correct value for a weight on low-level features such as an edge histogram. Additionally, users do not know about the underlying algorithms evaluating their query or the distance functions that are used to calculate the similarity of a media object to the query. Thus, the user is likely to set up a mental model [17,18] that does not sufficiently match the conceptual model of the application.

Besides this problem, condition weighting is also dynamic during the search. Acceptable media objects for the query depend on the user's mental image which is adjusted during the process because of two main reasons. First, users do not know the media repository from the beginning and are, secondly, likely to change their search goal in terms of similarity. Hence, subjective weights on conditions need continuous adjustment.

Our approach relies on user-defined posets to address the issues of dynamic condition weighting and the users' aversion to reviewing a large number of result objects.

2.1 The Preference Metaphor for User Interaction

Our approach can be considered a specialized relevance feedback method [1]. After an initial run of the system (see section 3 for details about the initial weighting) the user is given the chance to review listed results found by the retrieval system. Traditional relevance feedback approaches allow rating the result objects with respect to a query

[5] i.e. a variety of low-level feature conditions.

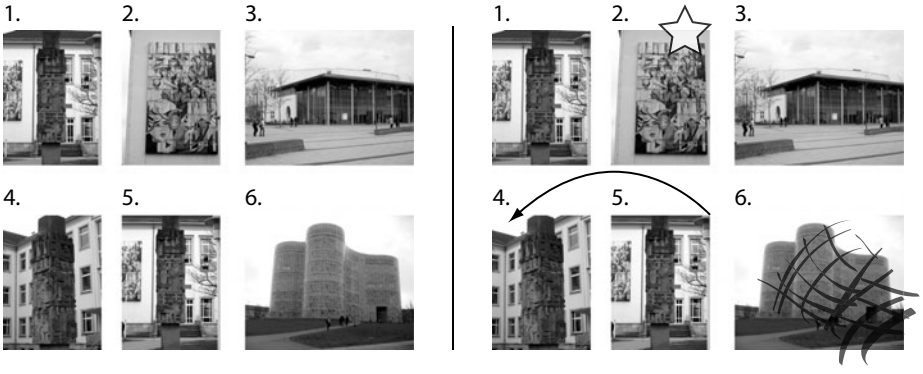


Fig. 2. Building posets; left: original result set R, right: user-defined poset P

while our system allows the indirect creation of a poset[6] over the result set R[7] (see Fig. 2). This enables the user to express relations such as 'more relevant than' for certain pairs from R. Particularly, it is not necessary to review all result objects as the system uses the user-defined poset to learn the user's subjective perception of relevance or similarity to calculate a total order using condition weights for the given query (see section 4).

Figure 2 depicts an ordered result set $R = \{o_1, o_2, o_3, o_4, o_5, o_6\}$ that has been generated according to the query shown in figure 1. The right side of the figure shows the poset P; $P \subseteq R \times R$ defined by the user according to his preferences. In order to facilitate the creation of the needed poset from a user's perspective additional metaphors have to be introduced.

Favorite objects: Users can mark several objects from R as their favorites (see the $\star$-marked o_2 at fig. 1). This means for the poset that the following holds:

$$\forall\, o \in R : (o_\star \geq o) \in P$$

Preference arrangement: Users can arrange the order of two objects from R to express their different reception of similarity or relevance for the two elements (see Fig. 1: o_4 and o_5).

Trash bin: Objects can be moved to a trash bin in case they are not relevant to the query. This gives the elements with the lowest relevance:

$$\forall\, o \in R : (o_\dagger \leq o) \in P$$

Pinboard: An additional pinboard should be provided to save objects that are not directly relevant to the query but might become useful to the user in later tasks.

[6] A partially ordered set is a set with a partial order expressed by a reflexive, antisymmetric, and transitive binary relation $\leq$ for certain element pairs that denotes their order.

[7] Totally ordered in terms of similarity to the query, i.e. list semantics.

It is obvious that more metaphors are used in the user interface than necessary from a theoretical point of view. This makes the interaction more intuitive in comparison to one global poset because metaphors such as favorites, preferences, and trash bins are already familiar to the user. The definition of favorite or irrelevant objects only using a binary ordering relation is far too laborious. Note that we do not depend on the declaration of parts of the poset that can be derived by reflexivity or transitivity. The pinboard is provided for convenience to allow parallel search as a benefit from the iterative process and has no influence on the poset.

Figure 3 shows one possible interpretation of the user's preferences neglecting the list semantics of the original result set R. The user has chosen a small number of favorite objects ($o_\star$), rearranged some objects (o_4 and o_5), removed objects (o_6), and has not interacted with some objects at all (see the pyramid's base). It can be estimated that the last group forms the majority due to the user's disinterest in extensive interaction with the system [19]. Objects that have not been reviewed by the user form the second lowest group within the poset having only more relevance than the objects in the trash bin.

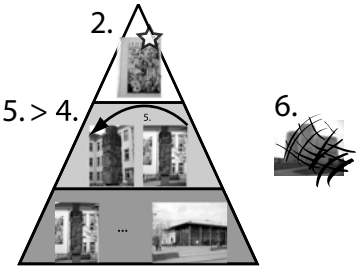


Fig. 3. Estimated distribution of objects in the poset

Another approach exploits the list semantics of the original result set R. Given a list ordered by relevance, the user only rearranges the objects that do not conform his perception of relevance in addition to his choice of favorite and irrelevant results. The main difference to the first approach is that objects, which have not been touched by the user, keep their position in the list and are therefore included in the derived poset according to their original order. While the second approach seems tempting, we propose the first alternative which takes only result objects into account that have been part of the user interaction. It can be assumed that users, who try to avoid superfluous interaction, will not rearrange objects they are not interested in. These objects will be placed in a significantly lower position in the poset in comparison to the second approach where the users' laziness is going to 'reward' highly ranked results that are not relevant to the user.

2.2 Basic Requirements of a Supportive Learning Algorithm

The subjective concept of similarity and the dynamic search process poses two main challenges for a supportive learning algorithm. First, the system has to adjust to the user's perception of relevance and similarity with respect to a given query. As discussed before, this can be achieved by an iterative process using interactively defined

posets over result sets that are used to calculate a weighted query according to the user preferences. This poses an immediate problem. Due to the nature of our non-restrictive user interaction the user can formulate posets that are in conflict with the given query. In this case, total orders may not be derived from the given poset and query. To address this issue, the learning algorithm has to check user-defined posets against the given query in order to reveal possible conflicts[8]. Conflicts can be discovered by an automatic verification of correct poset characteristics such as transitivity or the like. Consider the following example of a conflict: the user creates a "'poset"' with the following ordered pairs $A > B$, $B > C$ and $C > A$ which violates transitivity. In addition, conflicts can be revealed if the optimization algorithm (see Sec. 4) fails finding condition weights for a given query and poset showing that the query is not adequate.

If the evaluation discovers a conflict, it has to be traced to show to the user the contradictory ordering relations, so the user can decide the next step. One possibility is to let the user resolve the problem manually e.g. by deselecting conflicting ordering relations. The other is to let the algorithm choose to skip conflicting parts of the poset to ensure at least a maximized possible solution with less constraints. Conflicting parts can be discovered by a brute force approach testing all subsets of the provided poset against the results of the optimization of the given query. Since a small number of preferences can be expected due to the user's laziness, this approach works efficiently. To facilitate the interaction with the system the described evaluation has to run parallel to the user's poset creation. This leads to less manual correction in comparison to a serial evaluation process as conflicts are discovered during the review and can be resolved immediately. Finally, the conflict-free poset in addition to the given query form the input for a learning algorithm that will be discussed in section 4.

Secondly, the system has to adjust to the dynamic search goal of the user. As said previously, users may change their initial search goal when they gain insight into the contents of a media repository or when they become clearer about their actual understanding of similarity. It is important to identify moments when the user goal changes to ensure a new start for the learning algorithm. If the algorithm would not readapt to the new user goal it would divide the user's and the system's 'understanding' of relevance and similarity to the given query resulting in a mismatch in expected and delivered results. In the worst case, the users would try to adapt to the system based on a incorrect mental model lowering the quality of the results even more.

3 Initial Weighting Approaches

As mentioned before, initial condition weighting plays a special role for a user-satisfying relevance feedback process. Based on initial weights, first results are generated and shown to the user. Along with all other weighted queries during the entire process, the results are compared to the users' subjective perception of similarity and their mental image of an optimal search result. The weights build therefore the basis for further interaction with the system. In order to model the concept of subjective perception of

[8] Conflicts occur if a poset can not be derived from the user input or if the user models situations such $A \; covers \; B$ and $B \geq A; A, B \in R$, while A fulfills all conditions better than B and the given query does not contain a negation.

relevance in terms of similarity based on certain key features the following six initial weighting opportunities will be discussed.

1. *Weighting based on explicitly or implicitly user-defined conditions*
   This approach exploits semantic information from the user's input mode. It takes into account that some conditions[9] are defined explicitly by the user. Other conditions, especially dealing with low-level features, are input implicitly by providing a sample image and not by the provision of certain low-level feature sets. Based on this finding, explicit input conditions gain more influence on the actual query weighting in comparison to implicit conditions by mapping these relative constraints to absolute weighting values. While this assumption may hold for a majority of use cases, there are few examples where users are, for example, too lazy to input low-level features explicitly although the direct similarity to a media object is more important than a certain date of creation.
2. *Weighting according to search profiles or patterns*
   Initial condition weights are set on base of search profiles, e.g. the search after engravings. Typical characteristics of an engraving are texture and edges while colors can be neglected. The usage of search profiles assists users in specific domain research but does neither provide means for a broad, unspecific search nor subjective user preferences.
3. *Weighting based on user profiles*
   In case that experiences about the user's perception of similarity are available, the system can use these facts to pre-adjust all condition weights to orientate itself along the user's preferences. This ensures a good adaptivity to the user's general preferences but lacks support for specific search tasks that can be handled with approach no. 2.
4. *Explicit user input of weights*
   The traditional approach uses condition weights that are directly controlled by the user. Note that we do not discriminate between direct input via text fields, etc., or GUI widgets such as sliders. While the latter is considered much more user-friendly, both input mechanisms have in common that they assume deep knowledge of the underlying algorithms, i.e. feature extraction, distance calculation etc., to ensure a reasonable operation by users which cannot be guaranteed.
5. *Neutral weights*
   If no prior knowledge about the user's perception or search profile is available, conservative neutral condition weighting can be used to return a large, unfocused result set. Note that this is equivalent to the assumption of equal important stimuli resulting in a similarity perception contradictory to research results from both psychology [6,7] and IR [8,9].
6. *Randomized weights*
   Randomized weights can be chosen to guess subjective preferences of the user. Although this approach generates more differentiated results than neutral weighting it risks to be far away from user expectations. This will lead to an additional workload for users as they have to rearrange the results according to their individual preferences.

[9] e.g. DB conditions.

In our opinion, only the first three approaches can be recommended. Especially, the first initial weighting scheme is useful in general research tasks and can be used directly without prior knowledge. It generates reasonably good results by exploiting additional informations from the user interaction that can only be exceeded by the domain or user specific profile approaches.

4 Algorithm

In our example query, see figure 1, we combine four conditions: one similarity condition, two database conditions, and one spatial proximity condition. The database query returns for every media object a Boolean value, whereas the remaining conditions return values, often called *relevance scores*, out of the interval $[0, 1]$. Unfortunately, traditional database technology is unable to deal adequately with relevance scores in conjunctions or disjunctions.

One solution attempt is the use of fuzzy logic as discussed in [20,21]. As a generalization of Boolean logic, it supports membership values interpretable as vague truth values together with a formalism to construct and evaluate complex conditions. However, with respect to our scenario, we are faced with a dilemma if we state the following requirements chosen to be near to a human comprehension of search:

1. The laws of Boolean algebra should be supported.
2. The result of a conjunction/disjunction of conditions on different properties should be sensitive to changes of *both* input truth values.

Fulfilling the first requirement provides the standard t-norm/t-conorm `min/max` for conjunction and disjunction which unfortunately does not satisfy the second requirement. The minimum, for example, is insensitive to changes of the larger input value. On the other side, any other t-norm/t-conorm fulfilling the second requirement violates the laws of a Boolean algebra.

A framework unifying database and retrieval search should meet both requirements. Our approach is to use results from quantum mechanics and logic in order to achieve our goal. The main idea is to use the algebraic product for conjunction, the algebraic sum for disjunction, and the subtraction from one for a negation only for queries in a particular syntactical form (disjointness) and only if for every attribute at most one non-database condition[10] is defined. In [11] we describe an algorithm based on the disjunctive normal form which uses rules from Boolean logic to transform any query into the required syntactical form. The following example demonstrates the transformation of overlapping conditions into disjoint conditions where a, b, c are atomic conditions on different attribute:

$$(a \wedge b) \vee (a \wedge c) \Rightarrow a \wedge (b \vee c)$$

Now we will sketch our query language CQQL (*commuting quantum query language*). For a given set of attributes we define a set AC of atomic conditions. An atomic condition is one of the following alternatives:

[10] This does not mean a restriction for retrieval queries based on different search terms since terms are there expressed by different dimensions of a chosen vector space model.

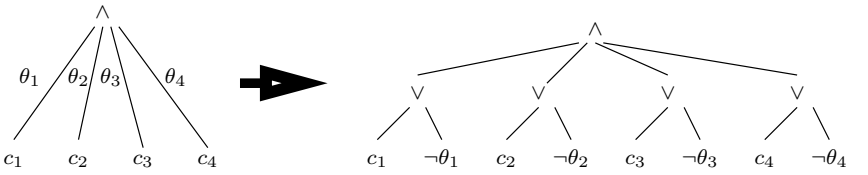


Fig. 4. Weight substitution in CQQL

1. A `select` condition '$A = c$' on an attribute A and a constant c is an atom.
2. An equality condition '$A_{i_1} = \ldots = A_{i_k}$' on k attributes of the same type is an atom.
3. A test on set containment of an attribute '$A \in C$' is an atom.

Every atomic condition can be a database or a non-database condition. Database conditions return a score value from $\{0, 1\}$ whereas the remaining ones[11] yield a score from $[0, 1]$.

A set of atomic conditions AC is called *commuting*, if no pair of non-database atoms restrict the same attribute differently. Every atomic condition is itself a CQQL query. By freely applying negation, conjunction, and disjunction and adding quantifiers we obtain complex CQQL queries.

Next, we equip our language with a weighting schema following [12]. The operands of a conjunction or a disjunction can be given weights from the interval $[0, 1]$ influencing the operand's impact on the result. A zero-weighted operand has no impact at all. Two one-weighted operands behave like unweighted operands. In contrast to many other weighting schemes, our weighting schema is linear and completely expressed in a Boolean logic:

$$\wedge_{\theta_1,\theta_2}(\varphi_1, \varphi_2) \Longrightarrow (\varphi_1 \vee \neg\theta_1) \wedge (\varphi_2 \vee \neg\theta_2)$$
$$\vee_{\theta_1,\theta_2}(\varphi_1, \varphi_2) \Longrightarrow (\varphi_1 \wedge \theta_1) \vee (\varphi_2 \wedge \theta_2)$$

φ_i denotes a score value as evaluation result and θ_i its respective weight. Thus, weighted CQQL queries are transformed to their unweighted counterparts and then evaluated as described above. Figure 4 illustrates the substitution of our example, where the four conditions are abbreviated by c_1, c_2, c_3, c_4.

Without restricting generality, let us assume that after the logical transformation the query condition is in the disjunctive normal form where every pair of different conjunctions contains at least one atomic condition which is negated on one side and not negated on the other side. The arithmetical evaluation [12] replaces any conjunction by a multiplication and any disjunction by an addition providing a formula of the form $\sum_{i=1}^{m} \left(\Pi_{j=i_1}^{i_n} x_i^j \right)$ where x_i^j is a non-negated expression e_i^j or a negated expression $\neg e_i^j$ evaluated by $1 - e_i^j$. Every expression e_i^j is an atomic condition $y_j^j(o)$ producing a score value for a media object o or a weight $\theta \in [0, 1]$.

For our learning algorithm we consider the weights θ as variables to be learned. Let us see what one preference relation $o_1 \geq o_2 \in P \subseteq R \times R$ means to our arithmetic

[11] Proximity, similarity, or retrieval conditions.

evaluation formula. Obviously, both objects o_1 and o_2 are evaluated by the same given CQQL query. That is, every atomic condition y_i^j on o_1 or o_2 yields a score value which together results in an inequation $\sum_{i=1}^{m} \varphi_i(o_1) * \Pi_{k=i_1}^{i_l} x_k \geq \sum_{i=1}^{m} \varphi_i(o_2) * \Pi_{k=i_1}^{i_l} x_k$ being equivalent to $\sum_{i=1}^{m} (\varphi_i(o_1) - \varphi_i(o_2)) * \Pi_{k=i_1}^{i_l} x_k \geq 0$. The expressions $\varphi_i(o)$ denote score values from evaluating atomic conditions for object o as factor of a summand whereas $x_k = \theta_k$ denotes the weight variable in the positive case and $x_k = 1 - \theta_k$ in the negative case.

The poset P contains a number of single preference relations. For computational simplicity, we assume, that P contains only non-reflexive preferences and preferences which cannot be derived transitively from other ones. For all preference relations we require a value greater than or equal to zero:

$$\min_{(o_1,o_2)\in P} \sum_{i=1}^{m} (\varphi_i(o_1) - \varphi_i(o_2)) * \Pi_{k=i_1}^{i_l} x_k \geq 0.$$

The learning problem is now to find weight values $\theta_k \in [0, 1]$ which satisfy that condition. One special case is the case when all weights are equal. In this case, we obtain a non-linear problem. Thus, the general problem is non-linear and can be considered as an optimization problem: *Which weights produce the largest value and is this value greater than zero?* Unfortunately, computing the maximum for non-linear equations is a computational hard problem.

As a learning algorithm we propose the downhill simplex algorithm from Nelder and Mead [22]. It starts from an initial setting for the weight variables and tries to find the maximum. However, it is possible, that the algorithm stops within a local maxima. Therefore, we run the algorithm several times with randomly chosen start values which incorporate the initial weights discussed in section 3.

We implemented our algorithm in Java and ran different experiments. Our goal was to find out how many preferences are needed in order to learn a total order. Therefore, we started with a simple conjunctive weighted query with given n weight values on a given object set. The evaluation produced a sequence of objects from which we were interested only in the first k objects. Next, we generated all possible preferences among the k objects. We found out that in most cases we need at least n preferences in order to learn n weights. Interestingly, this effect is independent from k. That is, learning weights does not need too many preferences. Furthermore, learning weights, if the number of weights and the number of preference is less than 20 is performed in less than a second on a normal PC.

5 Conclusion

This work presents a new relevance feedback approach that is based on the interactive creation of posets in order to adapt to the user's individual perception of similarity. A machine based learning algorithm is used to support users in order to present better result on base of a weighted query that correlates with their subjective preferences. The usage of a poset as input for such an algorithm is reasonable because of the difficulty of manual condition weighting. Our approach interprets the user's understanding of

similarity to a given query by exploiting posets that are used as a starting point for an automatic derivation of specific condition weights.

We plan to extend our approach into the field of logical connector learning to interpret queries conditions that are not connected with dis- or conjunctions. Additionally, the mechanism has to be integrated into a prototypical system to evaluate our results in a real-life environment. First result for a content management system used by the city of Cottbus (*cottbus.de*) look promising, but have to be strengthened by future experiments involving an additional prototype for multimedia search and further user evaluation.

References

1. Salton, G., Buckley, C.: Improving Retrieval Performance by Relevance Feedback. Technical report, Ithaca, NY, USA (1988)
2. van Rijsbergen, C.J.: Information Retrieval. Butterworths, London (1979)
3. Codd, E.F.: A Database Sublanguage Founded on the Relational Calculus. In: SIGFIDET (ed.) ACM SIGFIDET Workshop on Data Description, Access and Control, pp. 35–61 (1971)
4. Date, C.J., Darwen, H.: A Guide to the SQL Standard, 3rd edn. Addison-Wesley, Reading (1993)
5. Zadeh, L.A.: Fuzzy Sets. Information and Control (8), 338–353 (1965)
6. Bruce, V., Green, P.R.: Visual Perception –physiology, psychology and ecology, 2nd edn. reprinted.Lawrence Erlbaum Associates Publishers, Hove and London (1993)
7. Selfridge, O.G.: Pandemonium. A paradigm for learning. The mechanics of thought processes (1959)
8. Salton, G., Fox, E.A., Wu, H.: Extended Boolean Information Retrieval. Commun. ACM 26(11), 1022–1036 (1983)
9. Lee, J.H.: Properties of Extended Boolean Models in Information Retrieval. In: SIGIR (ed.) SIGIR 1994: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 182–190. Springer–Verlag New York, Inc. (1994)
10. Birkhoff, G., von Neumann, J.: The Logic of Quantum Mechanics. Annals of Mathematics 37, 823–843 (1936)
11. Schmitt, I.: QQL: A DB&IR Query Language. The VLDB Journal 17(1), 39–56 (2008)
12. Schmitt, I.: Weighting in CQQL. Technical Report 4, Cottbus (2007)
13. Weikum, G.: DB&IR: both sides now. In: SIGMOD (ed.) SIGMOD 2007: Proceedings of the 2007 ACM SIGMOD international conference on Management of data, pp. 25–30. ACM, New York (2007)
14. Rowe, L.A., Jain, R.: ACM SIGMM retreat report on future directions in multimedia research. ACM Trans. Multimedia Comput. Commun. Appl. 1(1), 3–13 (2005)
15. Lowell Workshop: The Lowell Database Research Self Assessment. Technical report (2003)
16. Fagin, R., Wimmers, E.L.: A Formula for Incorporating Weights into Scoring Rules. Special Issue of Theoretical Computer Science (239), 309–338 (2000)
17. Craik, K.J.W.: The Nature of Explanation. Cambridge University Press, Cambridge (1943)
18. Preece, J., Rogers, Y., Sharp, H.: Interaction design: Beyond human–computer interaction. Wiley, New York (2002)
19. Shneiderman, B., Plaisant, C.: Designing the user interface: Strategies for effective human–computer interaction, 4th edn. Pearson, Boston (2005)

20. Ciaccia, P., Montesi, D., Penzo, W., Trombetta, A.: Imprecision and User Preferences in Multimedia Queries: A Generic Algebraic Approach. In: Schewe, K.-D., Thalheim, B. (eds.) FoIKS 2000. LNCS, vol. 1762, pp. 50–71. Springer, Heidelberg (2000)
21. Schmitt, I., Schulz, N.: Similarity Relational Calculus and its Reduction to a Similarity Algebra. In: Seipel, D., Turull-Torres, J.M.a. (eds.) FoIKS 2004. LNCS, vol. 2942, pp. 252–272. Springer, Heidelberg (2004)
22. Nelder, J.A., Mead, R.: A simplex method for function minimization. Computer Journal 7, 308–313 (1965)

Adaptive User Modeling for Content-Based Music Retrieval

Kay Wolter[1], Christoph Bastuck[2], and Daniel Gärtner[3]

Fraunhofer Institute for Digital Media Technology, Ilmenau, Germany
www.idmt.fraunhofer.de
{woltky,bsk,gtr}@idmt.fraunhofer.de

Abstract. An approach to adapt a content-based music retrieval system (CBMR system) to the user is presented and evaluated. Accepted and rejected songs are gathered to extract the user's preferences. To compare acoustic characteristics of music files, profiles are introduced. These are based on result lists. Each result list is created by a classifier and sorted accordingly to the similarity of the given seed song. To detect important characteristics, the accepted and rejected songs are clustered with k-means. A score for each candidate song is specified by the distance to the mean values of the obtained clusters. The songs are proposed by creating a playlist, which is sorted by the score. Songs accepted by the listener are used to query the CBMR system for new songs and thus extract additional profiles. It is shown that incorporating relevance feedback can significantly improve the quality of music recommendation. The L_2 distance is suitable to determine similarities between profiles of regarded songs. Introducing more than one query song during the recommendation process can further improve the quality.

1 Introduction

In the recent years, the way to acquire and share music has changed. New distribution channels like the internet extend conventional channels, due to the availability of broader bandwidth and techniques for audio compression. Music is now provided as downloads, which leads to new business models such as *music flatrate services*. The increasing amount of available music can hardly be handled with traditional methods. Metadata merely gives a coarse description of the content, assumed that this information is correct or existing at all. Common information retrieval techniques based on metadata feature the problem of introducing less known artists into their recommendation process. Metadata is rarely given for this content and relations to other artists are hardly to detect. To overcome these problems, content-based music retrieval (CBMR) models the acoustic characteristics of music and computes similarities between songs in the model space. Recommendation is realized by analyzing the content of audio files.

However, human perception of music similarity can differ from person to person and situation to situation. Presumably, a song is represented by characteristics the user likes and dislikes. To increase the recommendation quality for

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 40–52, 2010.

the user, his relevance feedback needs to be captured and incorporated into the recommendation process.

2 Related Work

In [1], relevance feedback is incorporated by training a support vector machine (SVM) with songs, provided by the user as training samples. The system then iterates between proposing new songs to the user and training a new classifier on the labeled songs. This process is denoted *active learning*. The labels of interest are the songs located near the decision boundary of the SVM. The assumption is that these songs are the most informative, i.e. they are presented to the user to receive new labels. In contrast, a passive learning scheme trains a classifier on a large pool of randomly selected labeled data. It is shown that by choosing intelligently training samples, the active learning approach requires half as many labels to achieve the results of the passive learning scheme or, alternately, can increase the precision-at-20 by ten percentage with the same number of labels.

In [2], songs the user likes and dislikes are gathered by analyzing the user's skipping behaviour. The user does not have to label songs explicitly. A pool of positive and negative rated songs is created. Songs the user completely listens to are rated positive, songs the user skips are rated negative. With regard to [3], nearest neighbor distances of candidate songs to the labeled songs are computed to provide the user new recommendations.

3 Concept

3.1 System Overview

An overview of the system components is given in Fig. 1. The *recommendation engine* (RE) creates a playlist sorted by the distance to a given song. This song is specified by the user and will be denoted as *seed song*. The *user feedback system* (UFB system) analyzes the user's relevance feedback about already played songs and adapts the playlist accordingly.

Audio features are extracted from the audio signal to capture relevant information about the musical characteristics. The models for each song are stored in a model database. In accordance to their semantic entropy, the audio features are divided into low-level, mid-level and high-level audio features. Low-level features describe abstract characteristics of single frames of an audio signal. The implemented features are spectral centroid, spectral flatness, spectral crest factor, audio spectrum envelope, Mel-frequency cepstral coefficients, loudness, and zero crossing rate. A detailed description can be found in [5]. Mid-level features are designed to describe abstract characteristics of the entire audio signal or segments. A straightforward approach to form mid-level features is computing the mean and standard deviation of time series of low-level features. Additionally, some techniques are described in [6]. High-level features are designed to provide semantic information such as full music annotation, denoted as aspect,

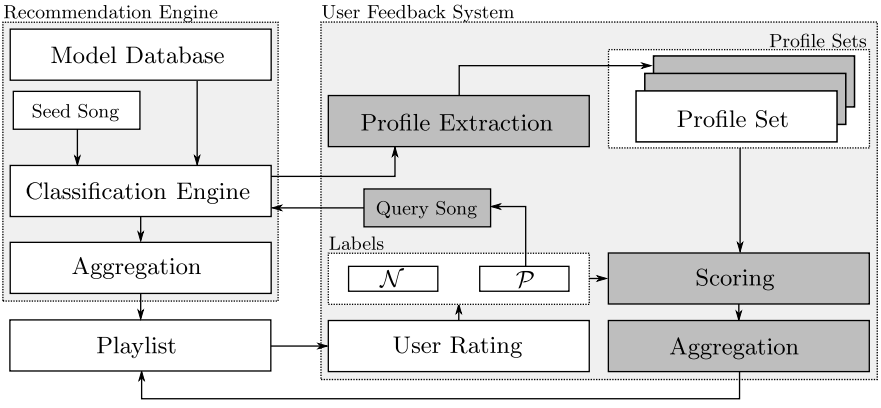


Fig. 1. System overview

or a transcription of the audio signal. A single aspect is created by a semantic classifier, which uses supervised learning on a given set of training samples ([7]).

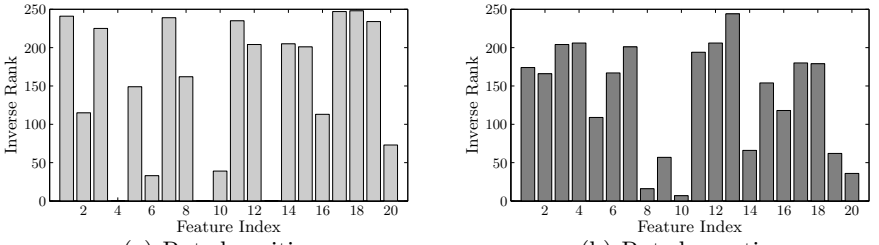
The classification engine computes relative distances of the models to the seed song. For each audio feature, a list is created with candidate songs, sorted according to the appropriate distance function. The final playlist is created by aggregating these lists. A comprehensive overview of aggregation techniques is given in [8]. For each candidate, the aggregation score is computed by considering the occurences within the top n ranks and the normalized mean rank in the lists, where the candidate occurs. The final playlist is sorted with respect to the obtained aggregation score.

Alternatively, the UFB system creates a playlist or adapts a given playlist, respectively. As proposed in [2], the system is designed in a manner to gather information about songs the user rated positive and negative, denoted $\mathcal{P}$ and $\mathcal{N}$. With regard to the results achieved with the active learning scheme discussed in [1], the system instantly reacts after receiving a new user rating and adapts the playlist. In the following sections, the components of the UFB system are described in detail. As an extension to the introduced approach, the highlighted parts are discussed in Section 3.4.

3.2 Profile Extraction

To analyze the importance of audio features, it is essential to reveal the arrangement of the lists from the classification engine. This is realized by extracting a profile for each song occuring at least in one list. Therefor, the classification engine is queried with a song to create a list for every audio feature. Regarding m audio features, a profile is a vector containing m elements, one for each list generated by the appropriate feature. An element A_k is proportional to the rank of the song in the corresponding list. However, A_k is computed as the *inverted rank*

$$A_k = n - R_k, \tag{1}$$



(a) Rated positive (b) Rated negative

Fig. 2. Profiles

with R_k as the rank in the list, if the observed song is within the top n ranks, otherwise $A_k = 0$. Large values for A_k indicate high similarity to the query song. The entire set of profiles, extracted with regard to a query song $\mathcal{Q}_j$, is combined to a *profile set* Π_j.

In Fig. 2, two profiles are presented computed with $n = 250$. The samples are rated differently, one positive (a), the other negative (b). The negative rated song is within the top ten ranks in the playlist. With regard to the structure of the according profile it is apparent that the song occurs in each of the considered lists, which leads to a large aggregation score. In contrast, the positive rated song shows high similarity to the seed song for a subset of elements and dissimilarities for others. It seems that various features give no meaningful contribution to discriminate between positive and negative rated songs. Hence, it is assumed that positive rated songs show high similarity to the seed song for the same subset of features. The profiles resemble each other, which is estimated by the L_2 distance.

3.3 Scoring

To adapt the recommendation to the user, the UFB system determines a score for each candidate song depending on the set of labeled songs $\mathcal{P}$ and $\mathcal{N}$. The processing can be summarized as follows: cluster the classes $\mathcal{P}$ and $\mathcal{N}$ and compute a score for each candidate song in $\mathcal{C}$. The new playlist is created by sorting the candidates with respect to their score.

The clustering is realized by k-means. For representing each class $\mathcal{P}$ and $\mathcal{N}$, k centroids are computed using the *Lloyd's algorithm* ([9],[10]). Training samples are profiles of the songs in $\mathcal{P}$ and $\mathcal{N}$, respectively. The set of clusters is denoted $Z_\mathcal{P}$ and $Z_\mathcal{N}$. For the profile of each candidate, the distances to all centroids of $Z_\mathcal{P}$ and $Z_\mathcal{N}$ are determined and afterwards weighted by the number of training samples assigned to the corresponding centroid. Let $n(j)$ be the number of training samples assigned to centroid $Z(j)$ and x_i be the profile vector of candidate $\mathcal{S}_i$, the score is

$$s_i = \frac{\sum_{j=1}^{k} n_{\mathcal{P}}(j)\ L_2(x_i, Z_{\mathcal{P}}(j))}{\sum_{j=1}^{k} n_{\mathcal{N}}(j)\ L_2(x_i, Z_{\mathcal{N}}(j))}, \tag{2}$$

with $L_2(x_i, Z_{\mathcal{P}}(j))$ as the euclidean distance of x_i to $Z_{\mathcal{P}}(j)$ and $L_2(x_i, Z_{\mathcal{N}}(j))$ as the euclidean distance of x_i to $Z_{\mathcal{N}}(j)$, respectively. The new playlist is generated by sorting the candidates ascending with respect to the obtained score.

3.4 Multi Profile Sets

Scoring creates a list for one profile set, generated by querying with one seed song. To capture more information about the user's preferences, positive rated songs are used as new query songs to generate additional profile sets. The assumption is that the extracted profile sets complement each other. The k-means scoring is used to detect relevant elements of the profiles of each profile set separately and thus to create a playlist for each profile set as described above. The components of this extension are highlighted in Fig. 1. All songs used to generate new profile sets are denoted *query song* $\mathcal{Q}_j$, whereas the first query song, which is determined by the user, is denoted *seed song* $\mathcal{Q}_1$.

The number of profile sets increases during the learning process. However, the system solely incorporates a determined number of N activated profile sets. For each of these profile sets Π_j, a list L_j is created. These lists are aggregated via mean rank score. Let $R_{i,j}$ be the rank of the observed candidate $\mathcal{S}_i$ in list L_j and N the number of activated profile sets, the corresponding aggregation score is

$$s_i = \frac{1}{\sum_{j=1}^{N} s_{i,j}}, \qquad \text{with } \begin{cases} s_{i,j} = \frac{1}{R_{i,j}}, \text{if } \mathcal{S}_i \in L_j \\ s_{i,j} = 0, \text{otherwise.} \end{cases} \tag{3}$$

Note that Eq. 3 is only applied for songs occuring at least in one list. Songs that do not satisfy this condition are not incorporated in the current playlist creation. The final playlist is created by sorting the candidates accordingly to their aggregation score.

It can be assumed that some profile sets are more suitable to represent the important characteristics than others. Furthermore, it seems reasonable that profile sets are exhausted after a certain number of recommendations. Additional positive recommendations are hardly to receive with these. An assessment of profile sets is made to reduce influences of profile sets that are inappropriate. Solely the profile sets with the top N assessments are incorporated.

The initial assessment for a new generated profile set is the number of profile sets temporary available. By this means, earlier generated sets give prior contribution to playlist generation. Subsequent to each user rating, an *update* of the assessments of the activated profile sets is made. This update depends on the contribution, the single profile sets gave to the recommendation of the actual

observed song. It is represented by the rank of that song in the corresponding list. Denoting the rank of the song in list L_j by R_j, the update for profile set Π_j is

$$\nu_j = \frac{\omega}{R_j}. \tag{4}$$

The factor ω regulates the rapidness of updates. Let ϑ_j be the assessment of the activated profile set Π_j, the assessment is updated by

$$\vartheta_j = \vartheta_j \pm \nu_j. \tag{5}$$

If the user rating of the observed song is positive, ν_j is subtracted from ϑ_j, otherwise ν_j is added. The more a profile set contributes to positive rated songs, the less is ϑ_j. The minimal assessment a profile set can have is 0. Thus, new profile sets are incorporated formerly, if an activated set is exhausted.

4 Experiments

4.1 Ground Truth

To evaluate the system, a ground truth needs to be defined which gives information about the recommendation quality. In [1], styles and moods are used for evaluation, categorized by All Music Guide (AMG). The styles are described as sub-genre, although AMG does not explicitly define them. In [11], Last.fm tags of individual artists are compared to evaluate a ground truth. It is shown that different users can associate the musical characteristics of the same artist with

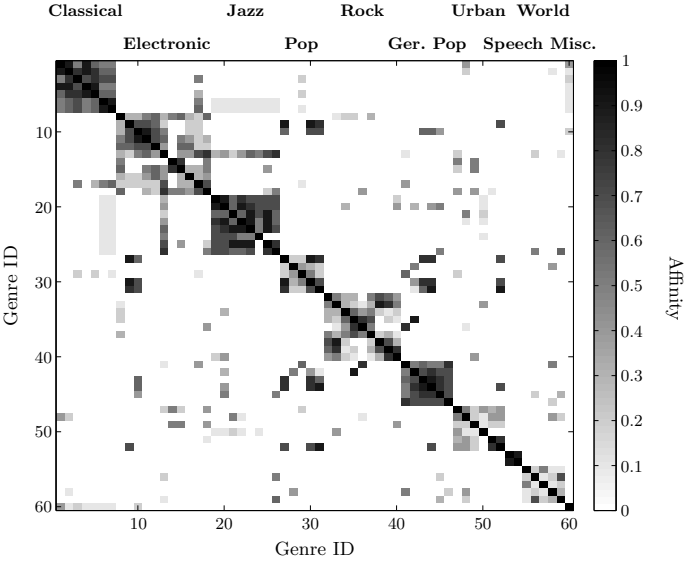


Fig. 3. Genre affinity matrix

Table 1. Results

Genre	#Songs	NAR			I_{NAR}[%]	
		RE	BASE	EXT	BASE	EXT
Classical	61	3.34	1.62	1.29	50.57	58.45
Electronic	170	4.31	3.43	2.29	21.22	39.93
Jazz	113	2.77	2.08	1.65	23.56	36.75
Pop	75	5.75	4.62	4.28	23.26	30.63
Rock	92	4.92	3.19	2.38	34.53	45.67
German Pop	54	3.68	3.00	2.59	18.24	28.55
Urban	110	8.22	5.69	3.65	27.62	46.96
Speech	30	3.55	1.95	1.13	43.48	53.61
Worldmusic	53	11.42	7.78	4.25	33.94	58.09
Miscellaneous	16	18.64	12.26	8.36	33.92	51.67
Entire Set	775	5.49	3.89	2.75	28.35	42.93

various genres. A range of affinities between genres are existing that have to be considered during the evaluation process.

A set of 775 songs is the database to evaluate the system. To estimate similarities between songs, sub-genre affinity is used as ground truth. The sub-genres and their affinities have been created during listening tests. The sub-genres are given in Table 2 and their mutual affinities are stored in a *genre affinity matrix* (Fig. 3). The affinities are in range $[0; 1]$, whereas 0 reflects dissimilarity and 1 indicates the affinity between identical sub-genres, i.e. high similarity, of the observed song.

4.2 Benchmark

The benchmark simulates a user's behaviour by querying the system with a seed song and "listening" to the created playlist. Similarity between recommendations

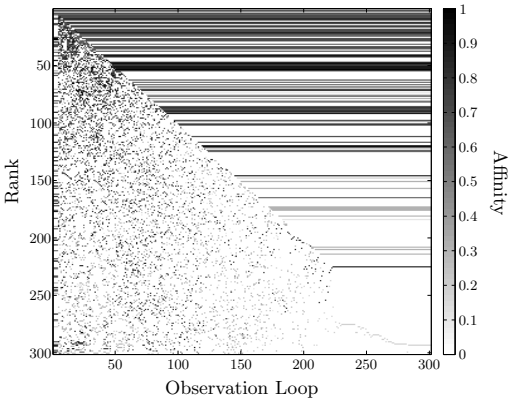


Fig. 4. Benchmark run, seed song: *Benny Moton - Rhumba Negro* (Dixieland Jazz), $\theta = 0.5$, list size 300

and the seed song is estimated by comparing the affinity between the sub-genre of an observed recommendation $\mathcal{S}_i$ to the sub-genre of the seed song. If the obtained affinity ρ_i is below a specified threshold θ, $\mathcal{S}_i$ is rejected (added to $\mathcal{N}$) and vice versa. Each rating, followed by the adaption of the list, is denoted *observation loop*. The rated songs are gathered by the UFB system. Based on these songs, the UFB system adapts the playlist by considering the remaining candidates and proposes the adapted list to the benchmark program. The described process is executed for a determined number of times. A complete run is illustrated in Fig. 4. The y-axis refers to the rank in the lists, obtained for the various observation loops (x-axis). The songs in the lists are represented by their affinity to the sub-genre of the seed song. By this means, one gets an estimate of the quality of a playlist. It is visible that the arrangement of the playlist varies depending on the number of observed songs. The list for observation loop 1 is the initial playlist, generated by the RE. The songs located on the main diagonal are the currently observed songs $\mathcal{S}_i$. The songs above have been already labeled and their arrangement does not further change. The songs below the main diagonal are the remaining candidate songs, sorted ascending with respect to the score computed by the UFB system.

4.3 Performance Metric

With regard to Fig. 4, the initial playlist (observation loop 1) is inferior to the playlist, produced by incorporating relevance feedback (observation loop 300). The dark points (positive songs) become more concentrated on the top of the playlist and the negative songs (white points) remain below. To express this impression, the *average rank measure* is applied for positive rated songs. However, this measure varies depending on the number of retrieved positive songs N. Furthermore, it can happen that the number of positive rated songs increases by incorporating relevance feedback. This leads to a decline of the average rank. Thus, the database is searched for the maximal number of positive songs, the system can retrieve, denoted $N_{\max}$. In the example given in of Fig. 4, the length of the playlist is 300, whereas the song set contains 775 songs. It is probable that positive songs are ranked below 300. To take these songs into consideration, their rank is approximated with the length of the playlist. Let S_L be the length of the observed playlist L, the summed ranks of the positive rated songs are

$$r_\Sigma = \sum_{i=1}^{N_{\max}} r(\mathcal{S}_i), \qquad \text{with} \begin{cases} r(\mathcal{S}_i) = \text{Rank of } \mathcal{S}_i \text{ in L, if } \mathcal{S}_i \in L \\ r(\mathcal{S}_i) = S_L, \text{otherwise} \end{cases} \tag{6}$$

and $\mathcal{S}_i$ as the currently observed positive rated song.

The best result is achieved, if all positive songs are on the top $N_{\max}$ ranks of the list, which can be expressed by the Gaussian formulation to compute the value of an arithmetic series:

$$r_{\text{Ref}} = \frac{N_{\max}(N_{\max}+1)}{2}. \tag{7}$$

Eq. (6) and (7) are set in relation to obtain the *normalized average rank*

$$\mathrm{NAR} = \frac{r_{\Sigma}}{r_{\mathrm{Ref}}}. \tag{8}$$

The top result is 1. This computation is made for the initial playlist L_I as well as for the adapted playlist L_A, produced by incorporating relevance feedback. Finally, the *improvement* is computed to estimate the performance of the UFB system. Let NAR_I be the NAR for L_I and NAR_A be the NAR for L_A, the improvement is

$$I_{\mathrm{NAR}} = \frac{\mathrm{NAR}_I - \mathrm{NAR}_A}{\max(\mathrm{NAR}_I, \mathrm{NAR}_A)} \cdot 100\%. \tag{9}$$

For the example in Fig. 4, $\mathrm{NAR}_I = 3.75$ and $\mathrm{NAR}_A = 2.87$, leading to an improvement $I_{\mathrm{NAR}} = 23.48\%$.

Furthermore, precision is used to illustrate the progression of the playlist quality with respect to a regarded rank r_L:

$$P(r_L) = \frac{n_{\mathcal{P}}}{r_L}, \tag{10}$$

with $n_{\mathcal{P}}$ as the number of positive songs above r_L.

4.4 Results

The results of the benchmark are presented for two configurations of the UFB system. Firstly, the UFB system incorporates solely the profile set extracted with the seed song. This is denoted the *basic system* (BASE). Secondly, the UFB system incorporates $N = 6$ activated profile sets, which is the *extended system* (EXT). The reference playlist to compute the improvement is the initial playlist, created by the recommendation engine (RE). The length of the playlists is $S_L = 400$. The number of centroids of the k-means clustering is limited to

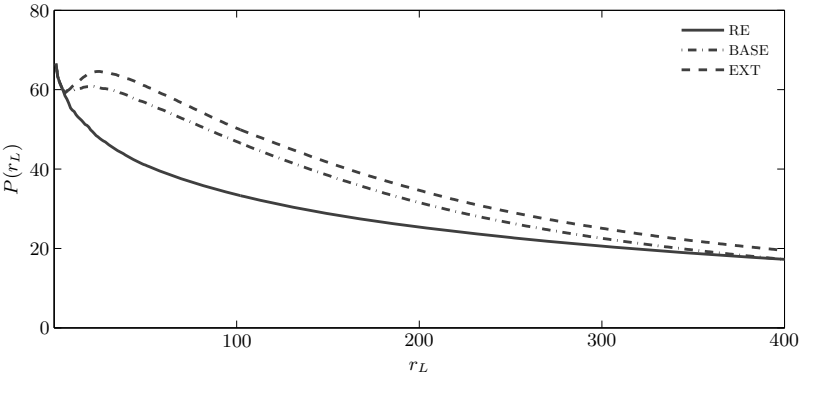


Fig. 5. Precision, extended system (EXT, $N = 6$) vs. basic system (BASE), processed with k-means, $k = 8$, averaged over the entire data set with every song as seed song

Table 2. Genre IDs

ID	Sub-Genre	# Songs
	Classical	
1	Choir	8
2	Art Song	11
3	Popular Classical	9
4	Opera	9
5	Ensemble	8
6	Guitar	8
7	Piano	8
	Electronic	
8	Big Beat	19
9	Hi -NRG Disco	10
10	Euro Dance	12
11	House	20
12	Techno	19
13	Lounge	29
14	Bright Drum&Bass	14
15	Dark Drum&Bass	10
16	Experimental	11
17	New Age	8
18	Trip Hop	18
	Jazz	
19	Dixieland	15
20	Swing	13
21	Blues	12
22	Entertainer	13
23	Bebop	31
24	Cool Jazz	9
25	Blue Note	10
26	Bossa Nova	10
	Pop	
27	Dance Pop	19
28	Country Music	8
29	Ballads	12
30	Italo Pop	10
31	70s Disco	26
	Rock	
32	Hard Rock	8
33	Nu Rock	21
34	Rock&Roll	8
35	Beat Music	13
36	Rock Ballads	8
37	Psychedelic	7
38	Metal	10
39	Punk/Hardcore	8
40	Ska	8
	German Pop	
41	German Ballads	8
42	German Beat	9
43	German Dance	9
44	German Disco	8
45	German Latin	8
46	German Traditional	12
	Urban	
47	Soul/RnB	20
48	Gospel	15
49	Reggae	14
50	Rap	19
51	Female Funk	19
52	Modern Funk	23
	Speech	
53	Speech without Sounds	19
54	Speech with Sounds	12
	Worldmusic	
55	Asian	13
56	Flamenco	10
57	Indian	11
58	Irish Ballads	11
59	Nubenegra	8
	Miscellaneous	
60	Children's Songs	16

$k = 8$ and depends on the number of training samples. Furthermore, songs with an affinity below 0.5 are rated negative, otherwise positive. The benchmark queries the RE with each song of the model database and thus initiates a run as illustrated in Figure 4. Subsequently, the run is evaluated by computing the NAR, improvement and precision. An estimate on the performance of the current configuration is given by the average over the entire data set of 775 songs.

In Figure 5, the precision of the playlists RE, BASE and EXT is displayed, depending on a considered rank r_L. The precision of the UFB lists and the RE list is equal for $r_L < 6$. The UFB system first has to gather a number of labels before adapting the playlist. Afterwards, the precision increases at about $r_L = 6$ and then declines at about $r_L = 20$. A remarkable improvement is obvious in comparison to the RE. This is also confirmed by the results in Table 1. The precision of the lists generated with the UFB system is throughout larger than the precision of the RE playlist. However, on $r_L = 400$, the precision of RE and BASE are equal, which is in contrast to the EXT playlist. Generally, the precision of the EXT playlist is larger than the precision of the BASE list. In Table 1, the

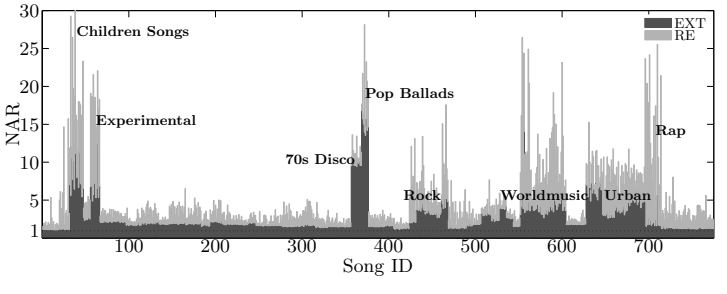


Fig. 6. Normalized average rank, recommendation engine (RE) vs. extended system (EXT, $N = 6$), processed with k-means, $k = 8$

improvement by incorporating the extended system ($I_{\mathrm{NAR}} = 42.93\%$) is 14.58% above the BASE playlist ($I_{\mathrm{NAR}} = 28.35\%$). In comparison to the RE playlist, the NAR is halved to 2.75. Hence, a positive rated song appears on average at each 2.75-th position in the playlist, assumed that there are remaining positive songs in the database.

In Figure 6, the NAR for the RE playlist and the EXT playlist are presented, obtained for each song of the database as seed song. Some genres and subgenres are marked, that previously led to problems by querying with these. It is shown that incorporating relevance feedback can essentially improve the quality of recommendation for these genres.

4.5 Discussion

As expected, the recommendation quality can be improved by incorporating relevance feedback. As shown in Figure 5, the system yields on average an improvement after the 6-th user rating. After reaching a maximum at about $r_L = 25$, the precision declines, which is presumably caused by the less number of similar songs in the test set. A larger number of positive songs in the database promises a deceleration of the decay.

Incorporating positive rated songs as new query songs further improves the quality. The activated profile sets appear to be complementary, since each represents important characteristics concerning the queried music style. In Figure 7, improvements are displayed, achieved depending on the number of activated profile sets (N). It is shown that increasing N results in higher performance of the UFB system. It seems surprising that the extended system with $N = 1$ performs superior to the basic system, although these appear to be alike. Hence, providing the option to reject unsuitable profile sets can essentially increase the quality. The system "samples" each profile set and retains solely the ones, that led to positive rated songs. With regard to Figure 5 it is obvious that the precision of the EXT playlist at $r_L = 400$ is above the precision of the RE playlist, which is not achieved for the BASE playlist. The system can reject a profile set, if it is

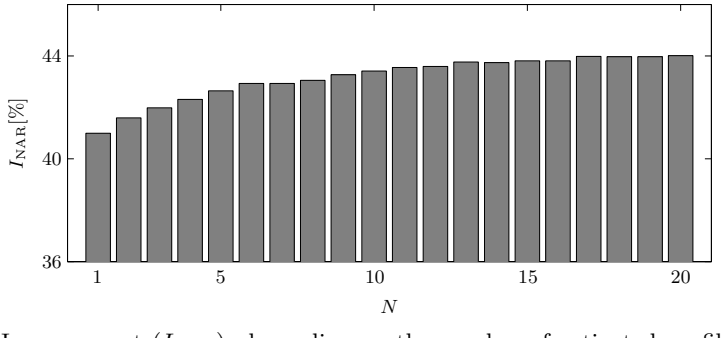


Fig. 7. Improvement (I_{NAR}), depending on the number of activated profile sets (N), processed with k-means, $k = 8$, averaged over the entire data set with every song as seed song

exhausted and thus retrieve additional positive songs, which are not present in the RE playlist as well as in the BASE playlist.

In view of Figure 6, the NARs computed for the RE playlists point out particular problems by proposing songs from the denoted genres and sub-genres. User relevance feedback can reduce these problems. For instance, the NAR of Rap is reduced from 13.31 to 2.38. However, the performance for 70s Disco and Pop Ballads cannot be improved significantly. It is assumed that the ground truth is inadequate for these sub-genres, since similarities to other sub-genres are hardly given in the affinity matrix (see Figure 3, Genre ID 28/29).

5 Conclusion and Outlook

A CBMR system is enhanced by incorporating the relevance feedback provided by a user into the recommendation process. It is shown that this information can be used to personalize the recommendation and thus improve the performance of the CBMR system with regard to a given scenario. Similarities between songs are determined by computing the L_2 distance between profiles, extracted for these songs. The results show an enhancement by using the k-means scoring to detect relevant elements of these profiles.

The quality increases significantly by introducing positive rated songs as new query songs. Hence, the characteristics of a queried music style are better represented by including various songs from that style. Since the quality and computing effort of the system increases by incorporating additional query songs, it can be adapted to the user's needs and the available computing power. Future work will focus on experiments with a larger number of songs. The system has to be verified on a database with millions of songs.

References

1. Mandel, M., Poliner, G., Ellis, D.: Support Vector Machine Active Learning for Music Retrieval. Multimedia Systems 12, 3–13 (2006)
2. Pampalk, E., Pohle, T., Widmer, G.: Dynamic Playlist Generation Based on Skipping Behaviour. In: International Conference on Music Information Retrieval, vol. 6, pp. 634–637 (2005)
3. Logan, B.: Music Recommendation From Song Sets. In: International Conference on Music Information Retrieval, vol. 5, pp. 425–428 (2004)
4. Lampropoulos, A., Sotiropoulos, D., Tsihrintzis, G.: Individualization of Music Similarity Preception via Feature Subset Selection. In: IEEE International Conference on Systems, Man & Cybernetics, vol. 1, pp. 552–556 (2004)
5. Peeters, G.: A Large Set of Audio Features for Sound Description (Similarity and Classification) in the CUIDADO Project. Technical Report, IRCAM, Paris, France (2004)
6. Dittmar, C., Bastuck, C., Gruhne, M.: Novel Mid-Level Audio Features for Music Similarity. In: International Conference on Music Communication Science, pp. 38–41 (2007)
7. Bastuck, C.: Weiterentwicklung eines Verfahrens zur automatischen Bestimmung musikalischer Ähnlichkeit. Master's Thesis, University Siegen (2006)

8. Dwork, D., Ravi, S., Naor, M., Sivakumar, D.: Rank Aggregation Methods for the Web. In: Proceedings of World Wide Web, vol. 10, pp. 613–622 (2001)
9. Kanungo, T., Mount, D., Netanyahu, N., Piatko, C., Silverman, R., Wu, A.: An Efficient k-Means Clustering Algorithm: Analysis and Implementation. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, 881–892 (2002)
10. Lloyd, S.: Least Squares Quantization in PCM. IEEE Transactions in Information Theory 28, 129–137 (1982)
11. Geleijnse, G., Schedl, M., Knees, P.: The Quest for Ground Truth in Musical Artist Tagging in the Social Web Era. In: International Conference on Music Information Retrieval, vol. 8, pp. 525–530 (2007)

Towards User-Adaptive Structuring and Organization of Music Collections

Sebastian Stober and Andreas Nürnberger

Data and Knowledge Engineering Group
Faculty of Computer Science
Otto-von-Guericke-University Magdeburg, D-39106 Magdeburg, Germany
{nuernb,stober}@iti.cs.uni-magdeburg.de

Abstract. We present a prototype system for organization and exploration of music archives that adapts to the user's way of structuring music collections. Initially, a growing self-organizing map is induced that clusters the music collection. The user has then the possibility to change the location of songs on the map by simple drag-and-drop actions. Each movement of a song causes a change in the underlying similarity measure based on a quadratic optimization scheme. As a result, the location of other songs is modified as well. Experiments simulating user interaction with the system show, that during this stepwise adaptation the similarity measure indeed converges to one that captures how the user compares songs. This utimately leads to an individually adapted presentation that is intuitively understandable to the user and thus eases access to the database.

1 Introduction

Automatic structuring is one means to ease access to music databases, be it for organization or for exploration. Of even greater help would be a presentation that adapts to the user's way of structuring music collections and thus is intuitively understandable. So far, aspects of individualization have been only a minor issue of research in the field of Music Information Retrieval (MIR) and Music Digital Libraries (MDL). Often, it is assumed that all users of a MIR/MDL system compare music in the same (objective) manner. This assumption, however, is usually not true: A musician, for instance, might especially look after structures, harmonics or instrumentation (possibly paying extra attention to his own instrument). Non-musicians will perhaps rely more on overall timbre or general mood when comparing songs. Others, in turn, may go in for the lyrics as long as they are able to understand the particular language. A MIR/MDL system incorporating this subjectiveness will better comply with the individual needs of its users and consequently gain a higher acceptance. Scenarios where this would be especially helpful are the exploration and organization of a music collection. Here, users would greatly benefit if a system would not just simply structure the collection for easier access but would structure it in a way that is intuitively understandable for the individual user by adapting to its preferences

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 53–65, 2010.

and needs. This, however, is not yet today's state of the art. At best, interfaces for music collection access allow for adaption by the user. However, they are lacking the ability to learn from user actions and to adapt on their own without explicit intervention of the user. In this paper, we present a prototype system that incorporates user adaptation in a simple yet effective manner.

In the following, we first give an overview on existing MIR/MDL systems that are either adaptable or adaptive as well as other related work sharing similar ideas. In Section 3 we describe our prototype system, specifically the used similarity facets and the adaptation method. Section 4 discusses the outcome of user simulations to objectively evaluate the adaptation method. Finally, we present ideas for future development in Section 5 and conclude with Section 6.

2 Related Work

Our approach builds upon the basic idea of adapting a similarity measures according to a user's preferences. This similarity can then be used to group songs in the collection and to generate a visual overview.

The idea of adapting similarity measures is not new: MPeer [3] allows to adjust the weight of three facets of music description in a similarity measure through an intuitive joystick interface for finding a set of similar songs given an anchor song.[1] The facets comprise the audio content, the lyrics and cultural metadata collected from the web. From a study with 10 users, it was concluded that users tend to use nearly similar joystick settings throughout different environments. Though the joystick interface is very intuitive, it is unclear whether it may be applied to more than 3 similarity facets. In the context of this paper, about 20 facets are used. Similarly, the E-Mu Jukebox [21] allows changing the similarity function that is applied to create a playlist from a seed song. Here, five similarity components (sound, tempo, mood, genre and year) are visually represented by adapters that can be dragged on a bull's eye. The closer a component is to the center, the higher is its weight in the similarity computation. This interface is scalable with respect to the number of facets but less intuitive. It may be hard for a user to explicitly specify a weighting scheme for the facets as this is usually something that only subconsciously exists. Especially with an increasing number of facets this is likely to become more difficult. Indeed, a user study with 22 participants showed that the users found the system harder to used but at the same time more useful compared to two control systems.

In contrast to the former systems, PATS (Personalized Automatic Track Selection) [13] is an adapative system for playlist generation that does not require manual adjustment of the underlying similarity measure but learns from user feedback. The system generates a playlist for a specific user context through dynamic clustering. The user can then select songs in the playlist that in his opinion do not fit for the current context-of-use. From this preference feedback, new feature weights in the underlying similarity measure are derived by an inductive learning algorithm based on the construction of a decision tree that

[1] For an online demo visit `http://mpeer.dfki.de`

uncovers the feature values classifying songs into the categories "preferred" and "rejected". Though the basic idea to learn from user feedback is indeed very similar to our approach, the usage scenario and the adaption algorithm are completely different: PATS (and the former systems) aims to generate lists of similar songs given one or more seed songs. In contrast to that, our goal is to structure a whole collection of songs. Hence, we do not classify songs as belonging or not belonging to a playlist but need to assign songs to cells of a self-organizing map (which can be regarded as a 1-of-n classification with each cell as a class).

Another adaptive system for playlist generation called PAPA (Psychology and Purpose-Aware Automatic Playlist Generation) [10] uses sensors that measure certain bio-signals (such as the pulse) as immediate feedback for the music currently played. This information is then used to learn which characteristics of music have a certain effect on the user. Based on this continuously adapting model playlists for certain purposes can be created. Though this method of getting immediate feedback for continuous adaptation is highly interesting, it is not applicable in the usage scenario of our approach.

For similarity-based grouping and visualization we use a growing self-organizing map (SOM) approach that has been already successfully applied for user-adaptive text, image and video retrieval [8,9,2]. Other MIR/MDL systems that utilize similar techniques comprise the SOM-enhanced Jukebox (SOMeJB) [14], the "Islands of Music" [12,11], the MusicMiner [6] and the PlaySOM- and PocketSOM-Player [7], the latter being a special interface for mobile devices. However, in contrast to our approach, none of these systems is based on an adaptive similarity measure.

3 Prototype

We have built a prototype that organizes music on song level to demonstrate our adaptation approach. The system is tested on the Beatles corpus containing 282 songs of The Beatles. Though this corpus is rather homogeneous compared to other collections cointaining songs of several artists and genres, it has some advantages: Firstly, the Beatles are well-studied and have already been a subject of MIR research for some time. Secondly, 180 songs (all songs from the 12 official albums) have been manually annotated with chord label at the Queen Mary University, London, and finally, there exists a vast amount of meta-data for the Beatles songs on the web. The following sections cover specific details of the system.

3.1 Features

For each song in the corpus, a set of features is extracted describing several facets of music similarity.[2]

[2] Throughout this paper, we use the term facet for a specific similarity that can be computed on a single feature or a combination of features. It is possible to derive several facets from the same feature(s) as different similarities may be computed as e.g. for the chord histogram. However, if not stated explicitly, there is always one facet/similarity for each feature.

Audio information: Audio data was available for 200 songs.[3] From these songs, audio features were extracted utilizing the capabilities of the frameworks CoMIRVA [17] and JAudio [5]. Currently, CoMIRVA supports two features representing the overall timbre as Gaussian Mixture Models of the Mel Frequency Cepstral Coefficients (MFCCs) according to [1] and [4] and a feature that describes how strong and fast beats are played within specific frequency bands [12]. These complex features have specific distance measures that are not bounded by 1. They were transformed into similarity measures with values in $(0, 1]$ as follows:

$$sim(a, b) = (dist(a, b) + 1)^{-1} \tag{1}$$

JAudio was used to extract a global audio descriptor as described in [19]. The similarity of the resulting 64-dimensional real number vectors that can be interpreted as a perceptual hashes is computed by using the euclidean distance.

Harmonic information: For the 180 songs from the official albums, the available ground truth chord labels were a basis for further feature computation.[4] From the chord labels, a histogram was built, mapping all chords onto the basic major and minor versions. For this histogram, two similarities were computed: Firstly, the cosine similarity from the respective (relative) chord frequency vectors, v_1 and v_2, defined as

$$sim(v_1, v_2) = \arccos \frac{v_1 \cdot v_2}{\|v_1\| \|v_2\|} \tag{2}$$

and secondly the Jaccard similarity of the sets of chords used (i.e. the non-empty histogram bins). For two sets, S_1 and S_2, it is defined as the size of the intersection divided by the size of the union:

$$sim(S_1, S_2) = \frac{S_1 \cap S_2}{S_1 \cup S_2} \tag{3}$$

As another feature, the tonal key was used as stated in Alan W. Pollack notes on The Beatles.[5] Again, the key is supposed to be automatically detected together with the chord labels in the next version of the system. The similarity for the key was computed with a modified version of the Jaccard similarity on the set of tones belonging to the key giving extra weight onto the more important ones.

Production information: As e.g. pointed out in [20], the production process plays an important role in defining the sound of a recording. We semi-automatically extracted such information from wikipedia[6] articles on the songs, comprising: creator(s) / composer(s), producer(s) and recording engineer(s),

[3] The audio data set comprises all songs from the 12 official albums, the Magical Mystery Tour EP, the Yellow Submarine soundtrack, and the "Blue', "Red" and "1" compilation.

[4] As such data is usually not available and can only be generated at high costs, we are currently incorporating an automatic chord recognizer into the system.

[5] `http://www.icce.rug.nl/~soundscapes/DATABASES/ AWP/awp-notes_on.shtml`

[6] `http://en.wikipedia.org`

recording location(s) (usually a studio), recording date(s), (guest) musicians and instruments. Additionally, the musicians were further divided regarding their instruments into subgroups for lead-vocals, background-vocals, guitars, bass and drums/percussion. All these features are set-based and compared by using the Jaccard similarity, except for the date where a more sophisticated similarity measure was used to differentiate between (partially or fully) overlapping, adjacent and independent time spans. Furthermore, the year of the first release was used as simple feature. For two values of the year, y_1 and y_2, the similarity is defined as:

$$sim(y_1, y_2) = \max\left\{0,\ 1 - \frac{|y_1 - y_2|}{3}\right\} \tag{4}$$

This resembles a fuzzy membership function for y_1 with a triangular shape [22].

Textual information: Lyrics for all songs were obtained through the web service of LyricWiki[7], filtered for stop words, stemmed and described by document vectors with TFxIDF term weights [16]. As similarity measure, the cosine similarity was used. In the same way, song and album titles were processed and compared.

Finally, album covers were obtained through web search and linked to the songs. However, they are currently only used for labeling as described in section 3.3.

3.2 Grouping

Based on the facet similarities described in the preeceding section, we can define a combined similarity measure that is computed as a weighted sum. The sum of all facet weights is always 1.0. Initially, all weights are equal and later subject to adaption by the method discussed in Section 3.4.

Using this intial similarity measure, songs are clustered by the growing self-organizing map approach presented in [9]. The algorithm starts with a small initial grid composed of hexagon cells, where each hexagon refers to a cluster and is represented by a randomly initialized prototype. Each song is then assigned to the most similar cluster in the grid resulting in an update of the respective prototype and to some extend of the surrounding prototypes (depending on a neighborhood function). Having all songs assigned to the grid, the inner cluster distance can be computed for each cluster. If it exceeds some predefined threshold, the respective cluster is split resulting in a grown map. This process is repeated until no more cells need to or can be split or an empty cell occurs (i.e. a cluster with no assigned songs). It results in a two-dimensional topology preserving the neighborhood relations of the high dimensional feature space. I.e. not only songs within the same cluster are similar to each other but also songs of clusters in the neighborhood are expected to be more similar than those in more distant clusters. Using a growing approach ensures that only as many clusters are created as are actually needed.

[7] `http://lyricwiki.org`

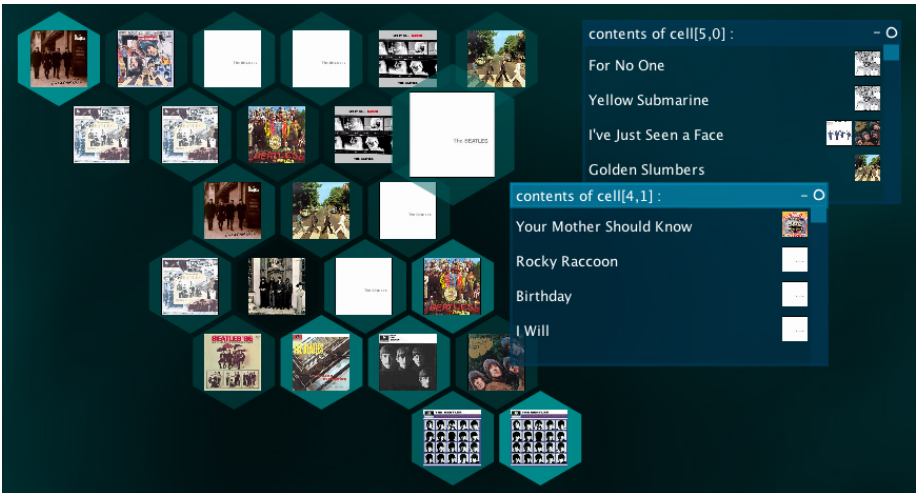


Fig. 1. Screenshot of the grid in *Cluster-Size-Mode* and two cell content windows

3.3 Presentation / User Interface

Having generated a hexagonal grid, it's cells can be seen as "virtual folders", each one containing a set of similar songs. A screenshot of the prototype user interface is shown in Fig. 1.[8] Cells are labelled with the album cover(s) that are most frequently linked with the songs contained in the cell. In the default view, only one cover is shown. At higher zoom level, at most four covers are displayed depending on how many different covers are linked with the songs in the cell. Further, the cell background can be colored providing different information:

1. Cluster-Size-Mode:
   The brightness of the cell color on an emerald color scale refers to the size of the respective cluster, where small cells are darker. This is the default mode (see Fig. 1).
2. Cluster-Error-Mode:
   This mode is by default chosen during the initial map learning phase. Cells are colored red with brightness referring to the internal cluster error, i.e. the sum of the distances of the prototype with all songs in the cluster.
3. No color:
   In this mode, the cells are not colored and only the covers are shown.

If the mouse is over a specific cell, this cell is visually enlarged and after some delay, a tooltip with a list of contained songs is displayed. Clicking on a cell opens a window that displays the content of the respective cluster. For each song, the title and (if available) the CD covers are listed.

[8] A demo video is available at `http://www.dke-research.de/aucoma/`

3.4 User-Adaptivity

The algorithm outlined in Section 3.2 works in an unsupervised manner. It depends only on the choice of the features for representation and the initial similarity measure how songs are grouped. However, the user has often a better intuition of a "correct" clustering. In cases where he does not agree on the cluster assignment for a specific song, he can change it by dragging the song from a cell's content window and drop it onto a different cell of the grid or another content window. Such an action does immediately result in a modification of the underlying facet weighting scheme and ultimately to a reassignment of all songs to the grid. This way, manually moving a single song may cause automatic movement of several others. The idea of the approach is that, while the user is interacting with the system nn an iterative process of user action and resulting adaptation, the similarity measure more and more converges towards to the (possibly only subconsciously existing) preferences of the user.

For the computation of the new facet weights, a quadratic optimization scheme is used that has been introduced in [18] where it was evaluated on text data. In the following, this method is briefly summarized. The formalization has been adapted to incorporate the concept of facet similarities as introduced in the context of this work.

The similarity $sim(a, b)$ of a pair of objects, a and b, is computed as:

$$sim(a, b) = \sum_{i=1}^{n} w_i \cdot sim_i(a, b) \tag{5}$$

where n is the number of facets, $sim_i(a, b)$ is the i-th facet similarity of a and b and w_i the respective facet weight. The facet weights are required to be non-negative and the sum of all weights has to be 1:

$$w_i \geq 0 \qquad \forall 1 \geq i \geq n \tag{6}$$

$$\sum_{i=1}^{n} w_i = 1 \tag{7}$$

The problem of dragging objects as described above can be mapped to a problem of cluster reassignement: Assume a object o that is assigned to a cluster c_s, i.e. for the similarity holds

$$sim(c_s, o) > sim(c, o) \qquad \forall c \neq c_s \tag{8}$$

If d is dragged by a user to a cluster c_t the underlying facet weights have to be adapted such that now

$$sim(c_t, o) > sim(c, o) \qquad \forall c \neq c_t \tag{9}$$

holds. If the user has already reassigned some object(s), a similar constraint has to be defined for each object to ensure that it stays at its position. Note

that the change of the facet weights should be as small as possible in order to avoid too abrupt changes of the cluster assignments. Therefore, it is additionally demanded that the sum over all (quadratic) deviations of the weights from their initial value $\frac{1}{n}$ should be minimal:

$$\min_{(w_1,\ldots,w_n)\in R^n} \sum_{i=1}^{n} \left(w_i - \frac{1}{n} \right)^2 \tag{10}$$

This can be used as the objective function for a quadratic optimization scheme and equations 6, 7 and 9 give the additional constraints.

Additionally to the above cluster reassignment scenario, we want to motivate another way of incorporating user feedback that can be as well integrated into the quadratic optimization scheme. The following scenario is considered: The user may query the system with a seed song. As a result, the system will display the 10 most similar songs to this seed according to its current weighting scheme. Form the user's point of view, this list may not be in the correct order. It may even not contain any of the songs the user would rank amongst the 10 most similar ones (if he would know all the songs in the collection). Consequently, the user may want to change the order of the songs.[9] From a single re-ranking of a song, additional constraints for the quadratic optimization can be derived as follows: Let s be the seed song, i be the original rank and $j \neq i$ be the modified rank of the re-ranked song d within the retrieved top-10 list. If j is less than i, it can be concluded that all songs with original rank $j \leq r < i$ are less similar than d and that the song at rank $j-1$ (if such exists) is more similar than d with respect to the seed song s. In the other case, i.e. if j is greater than i, all songs with original rank $i < r < j$ are more similar than d and the song originally at rank j is less similar than d with respect to the seed song s. Every relation "song a is more similar than song b with respect to seed song s" can then be expressed by the constraint:

$$sim(a, o) > sim(b, o) \tag{11}$$

Note the important difference with the constraint defined in Equation 10: Here, it is referred to objects instead of clusters.

4 Experiments

In order to objectively evaluate the usefulness of the system, we have simulated user interaction with the system as motivated and described in [18]. The basic idea is to assume a fixed random weighting scheme that the simulated user has "in mind". According to such a user model, some objects may be misplaced on the grid. Simulated interaction of the user should result in better adapted similarity measure of the system and thus in a decrease of the number of misplaced

[9] Such an action, e.g. through drag & drop on the ranked list, has not yet been incorporated into the graphical user interface. Currently, it can only be carried out through the simulation interface for evaluation.

objects. As proposed in [18], we use the *average top-10 precision* of the similarity measure for evaluation: A ranked top-10 list of similar objects is computed for each object according to the system's current similarity measure and compared with a top-10 list according to the preferences of the simulated user. The percentage of matches is computed – ignoring the order of the ranking – and averaged over all objects.

Apart from the average top-10 precision used in the earlier experiments, we also take the number of misplaced objects and the *object postion error (OPE)* as measures for the difficulty of the adaptation problem and the performance of the algorithm into account. The latter is defined as the euclidean distance of the currently assigned cluster $c(d) = \left(x_{c(d)}, y_{c(d)}\right)$ to the correct one $t(d) = \left(x_{t(d)}, y_{t(d)}\right)$ on the cell grid, summed over all objects d in the collection D, i.e.

$$OPE = \sum_{d \in D} \sqrt{\left(x_{t(d)} - x_{c(d)}\right)^2 + \left(y_{t(d)} - y_{c(d)}\right)^2} \tag{12}$$

We conducted two experiments – the first to verify that the adaptation approach through object repositioning is applicable in this domain, and the second to assess the usefulness of the newly introduced re-ranking constraints.

4.1 Experiment 1: Object Repositioning

In the first experiment, we measured the performance of the adaption method for a simulated user that moves misplaced objects according to his similarity measure. As modification of the simulations in [18], we used a simpler strategy for selection of the object to be moved: In each iteration, the set of misplaced objects is determined and one object of this set is randomly selected to be moved to its correct position. For the user's similarity measure, we consider the following three scenarios:

1. Fully random weights
2. Random binary weights (the weight vector is initialized as a random binary vector and normalized afterwards)
3. Single facet only (the weights for all facets except for a single random facet are zero)

Each scenario was tested on 10 self-organizing maps (learnt on the same data but with different random initializations) for 5 users (random weight vectors) and 5 simulations each (different objects are chosen to be moved manually). A simulation terminated when there were no more misplaced objects on the map. The results of this experiment are shown in Table 1.

The initial values for average top-10 precision, number of misplaced objects and object position error can be interpreted as indicators for the difficulty of the restructuring problem. The uniform weight vector initially assumed by the system and used for training of the map comes closest to a fully random weight vector (1). However, the average top-10 precision already drops by 30% for this scenario. It drops even further for the scenario using random binary vectors (2) and for single-facet weight vectors (3) it is only 11% resulting in a high number

Table 1. Results of the first experiment, simulating a user that moves misplaced objects. Mean (variance) computed over 250 random runs (10 maps × 5 users × 5 simulations) for each scenario of user weight initialization.

scenario (user weights)	(1) fully random	(2) random binary	(3) single facet
initial average top-10 precision	0.70 (0.01)	0.55 (0.01)	0.11 (0.01)
initially misplaced objects	4.87 (2.01)	11.42 (2.99)	239.63 (7.78)
initial object position error	7.33 (3.39)	17.97 (5.52)	638.32 (42.13)
final average top-10 precision	0.70 (0.01)	0.57 (0.02)	0.89 (0.04)
number of iterations	4.76 (1.94)	11.40 (2.94)	60.21 (11.08)

of misplaced objects (about 240 out of 282). For the other scenarios, (1) and (2), the number of initially misplaced objects is surprisingly small. However, as the final value for the average top-10 precision and the number of iterations show, there is not much improvement during the few iterations of scenarios (1) and (2). The few misplaced objects in these scenarios are almost completely moved manually by the user. The weight adaptations resulting from the simulated actions have not a big impact. For scenario (3) – which appears to be the hardest one – far more iterations are necessary but the final average top-10 precision is increased to 89% – the best value throughout all scenarios. Interestingly, in all scenarios the average top-10 precision would never come even close to 100%, even though all objects finally were at their correct position. The explanation for this "glass ceiling" effect lies in the nature of the self-organizing map. All objects are assigned to the correct cluster, as long as there is no other cluster that has a higher similarity (cf. equation 9). Regarding only constraints of this type[10], there may be many valid weighting schemes, the weighting scheme of the simulated user obviously being one of them. However, given only the user actions, any of the weighting schemes in the solution space could be the one of the user. The system has just not enough evidence, to decide which one is the right one, and chooses the one that minimizes the objective function (cf. equation 10). Especially with a large solution space, the chosen solution may significantly differ from the real one which results in the "glass ceiling" for the average top-10 precision. It is important to note, that this problem cannot be overcome by using a different adaptation strategy. As this effect was not evident in the experiment on text data in [18], it can be assumed, that because of the different nature of text data (i.e. high-dimensional but sparse vectors) and possibly because of the additional random data which was not added here, the solution space in these experiments must have been significantly smaller.

4.2 Experiment 2: Object Re-ranking

The second experiment aimed to assess the usefulness of the constraints derived from re-ranking a song within a retrieved top-10 list of the system. Note, that

[10] The other constrains given by equations 6, 7 can be neglected here as they can be met just by normalizing the weights.

using such constraints directly optimizes the average top-10 precision and does only indirectly reduce the number of misplaced objects through the adapted similarity measure. Consequently, a significantly higher number of iterations was necessary in these simulations, independently from the scenario of user weight initialization. An additional reason for this could be that the constraints derived from re-ranking may not contain as much information as the constraints from object repositioning – i.e. they may be not as restrictive. Thus, it takes longer for the system's facet weights to converge on the ones of the user. In the worst case (weight initialization as in scenario (1) of the first experiment), it took about 280 iterations until no more objects were misplaced. Figure 2 shows the average top-10 precision and the object position error for this simulation run. Interestingly, the average top-10 precision decreased during the first iterations – possibly mislead by the few constraints derived so far. Only after 46 iterations (244 constraints[11]) the baseline average top-10 precision of the initial solution (uniform weights) is crossed. An average top-10 precision of 99% is reached after 199 iterations (949 constraints). However, at this point, there are still 16 misplaced objects.

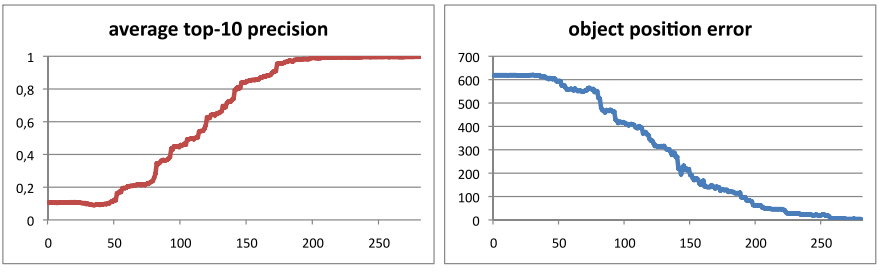


Fig. 2. Average top-10 precision and object position error for a simulation of adaptation through re-ranking constraints

As this experiment shows, using the re-ranking constraints allows better adaptation to the user. However this adaptation comes at significantly higher costs for the user because it requires far more manual actions. Combining both approaches – moving and re-ranking of objects – may be a way to benefit from their specific advantages. This has to be analyzed in further experiments, possibly with real users to find out, which actions are preferred and perceived as being easier.

5 Future Work

The prototype presented here is in an early stage of development and many aspects, especially concerning the automatic feature extraction leave room for

[11] As described in Section 3.4, multiple constraints in the form of equation 11 can be derived from re-ranking a single song.

improvement. An automatic chord recognizer based on the approach, we describe in [15] will soon be integrated into the system to replace manual annotations. For measuring the harmonicity of a song, possible further refinements are currently elaborated such as using the kurtosis ("peakiness") of an adequately aligned chord histogram as a measure for harmonic simplicity. Regarding the features currently used for facet similarity computation, there are large differences in complexity. The lyrics are for instance a very complex feature as opposed to the year. Possibly, facets referring to complex feature could be subdivided into sub-facets. This would lead to hierarchical similarity measures and pose new challenges for the adaptation method but at the same time possibly add more degrees of freedom. Regarding the initial weighting scheme, there may be much better initializations than just weighting all facets equally. Further, we are investigating other options of similarity based collection visualizations where the quadratic optimization method could also be applied.

6 Conclusion

We presented a prototype systems for structuring and exploring music collections that adapts to a user's way of comparing songs by learning from his interaction with the system. The system considers about 20 facets covering e.g. sound, harmonics, lyrics and information about the production process. These facets are combined in a complex similarity measure. For learning user specific facet weights, we extended an existing quadratic optimization approach that has previously been applied in the context of adaptive text retrieval. In a user simulation experiment, we verified that the system is able to approximate a similarity measure according to some unknown user preferences.

Acknowledgements

This work is supported by the German Research Foundation (DFG) and the German National Merit Foundation. We further would like to thank the developers of CoMIRVA [17] and JAudio [5] for providing their feature extractor code and George Tzanetakis for answering lots of questions concerning his MIREX 07 system [19]. Christopher Harte kindly shared his Beatles chord annotations with us.

References

1. Aucouturier, J.-J., Pachet, F.: Improving timbre similarity: How high is the sky? Journal of Negative Results in Speech and Audio Sciences 1(1) (2004)
2. Bärecke, T., Kijak, E., Nürnberger, A., Detyniecki, M.: Video navigation based on self-organizing maps. In: Sundaram, H., Naphade, M., Smith, J.R., Rui, Y. (eds.) CIVR 2006. LNCS, vol. 4071, pp. 340–349. Springer, Heidelberg (2006)
3. Baumann, S., Halloran, J.: An ecological approach to multimodal subjective music similarity perception. In: Proc. of CIM 2004 (2004)

4. Mandel, M., Ellis, D.: Song-level features and support vector machines for music classification. In: Proc. of ISMIR 2005 (2005)
5. McEnnis, D., McKay, C., Fujinaga, I., Depalle, P.: jAudio: An feature extraction library. In: Proc. of ISMIR 2005 (2005)
6. Mörchen, F., Ultsch, A., Nöcker, M., Stamm, C.: Databionic visualization of music collections according to perceptual distance. In: Proc. of ISMIR 2005 (2005)
7. Neumayer, R., Dittenbach, M., Rauber, A.: PlaySOM and PocketSOMPlayer, alternative interfaces to large music collections. In: Proc. of ISMIR 2005 (2005)
8. Nürnberger, A., Detyniecki, M.: Weighted self-organizing maps - incorporating user feedback. In: Artificial Neural Networks and Neural Information Processing - ICANN/ICONIP 2003, Proc. of the joined 13th Int. Conf. (2003)
9. Nürnberger, A., Klose, A.: Improving clustering and visualization of multimedia data using interactive user feedback. In: Proc. of the 9th Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU 2002 (2002)
10. Oliver, N., Kreger-Stickles, L.: PAPA: Psychology and purpose-aware automatic playlist generation. In: Proc. of ISMIR 2006 (2006)
11. Pampalk, E., Dixon, S., Widmer, G.: Exploring music collections by browsing different views. In: Proc. of ISMIR 2003 (2003)
12. Pampalk, E., Rauber, A., Merkl, D.: Content-based organization and visualization of music archives. In: Proc. of ACM MULTIMEDIA 2002 (2002)
13. Pauws, S., Eggen, B.: PATS: Realization and user evaluation of an automatic playlist generator. In: Proc. of ISMIR 2002 (2002)
14. Rauber, A., Pampalk, E., Merkl, D.: Using psycho-acoustic models and self-organizing maps to create a hierarchical structuring of music by musical styles. In: Proc. of ISMIR 2002 (2002)
15. Reinhard, J., Stober, S., Nürnberger, A.: Enhancing chord classification through neighbourhood histograms. In: Proc. of the 6th International Workshop on Content-Based Multimedia Indexing, CBMI 2008 (2008)
16. Salton, G., Buckley, C.: Term weighting approaches in automatic text retrieval. Information Processing & Management 24(5), 513–523 (1988)
17. Schedl, M.: The CoMIRVA Toolkit for Visualizing Music-Related Data. Technical report, Johannes Kepler University Linz (2006)
18. Stober, S., Nürnberger, A.: User modelling for interactive user-adaptive collection structuring. In: Boujemaa, N., Detyniecki, M., Nürnberger, A. (eds.) AMR 2007. LNCS, vol. 4918, pp. 95–108. Springer, Heidelberg (2008)
19. Tzanetakis, G.: Marsyas submission to MIREX 2007. In: Proc. of ISMIR 2007 (2007)
20. Tzanetakis, G., Jones, R., McNally, K.: Stereo panning features for classifying recording production style. In: Proc. of ISMIR 2007 (2007)
21. Vignoli, F., Pauws, S.: A music retrieval system based on user driven similarity and its evaluation. In: Proc. of ISMIR 2005 (2005)
22. Zadeh, L.A.: Fuzzy sets. Information and Control 8, 338–353 (1965)

An Approach to Automatically Tracking Music Preference on Mobile Players

Tim Pohle[1], Klaus Seyerlehner[1], and Gerhard Widmer[1,2]

[1] Department of Computational Perception
Johannes Kepler University Linz, Austria
[2] Austrian Research Institute
for Artificial Intelligence (OFAI)
Vienna, Austria
music@jku.at

Abstract. More and more music is being made available to the music listener today, while people have their favorite music on their mobile players. In this paper, we investigate an approach to automatically updating the music on the mobile player based on personal listening behavior. The aim is to automatically discard those pieces of music from the player the listener is fed up with, while new music is automatically selected from a large amount of available music. The source of new music could be a flat rate music delivery service, where the user pays a monthly fee to have access to a large amount of music. We assume a scenario where only a "skip" button is available to the user, which she presses when the currently playing track does not please her. We evaluate several algorithms and show that the best ones clearly outperform those with lower performance, while it remains open how much they can be improved further.

1 Introduction

Today, portable music players and mobile communication devices are more and more merging. For example, many mobile phones have audio players built into them. Thus, there is the possibility that people obtain new music by using a service over the air. The idea to offer such a service in the form of a music flat rate has been around for a while (be it over the Internet or immediately over the phone's data connection). Some flat rate music delivery services are being planned, and others are already available. For example, services are being offered by Yahoo[1], Microsoft[2], Napster[3], and diverse mobile communication providers, such as One[4]. The user pays a fixed monthly fee and can listen to music with little or no limitations with respect to the number of tracks. Some services include the option to transfer the tracks to mobile players for listening to them offline.

[1] http://music.yahoo.com
[2] http://music.msn.com
[3] http://www.napster.de
[4] http://www.one.at

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 66–77, 2010.

In such a scenario, it might be beneficial to support the user with MIR based techniques. Two aspects come to mind: First, the user may be supported in selecting those tracks she likes from all those that are available over the service. Second, as the amount of storage is eventually limited on any device, the user may be supported in selecting those tracks she wants to be deleted from the player (or moved to a larger persistent storage). The ultimate idea of such a MIR application is to keep the collection on the user's device up-to-date at any point in time, with only little effort (and/or interaction) by the user.

In this work, we present algorithms aimed to accomplish such a task based on an audio similarity measure [AP04, ME05]. Using user feedback to select matching tracks has been suggested in [PPW05, Pam06], from which we adopt ideas for track selection algorithms.

The remainder of this paper is organized as follows: After a brief literature review in Section 2, Section 3 gives a detailed and formalized description of the suggested application. The proposed algorithms are described in Section 4 and results presented in Section 5. Conclusions are drawn in Section 6.

2 Related Work

A number of MIR systems have been developed to support users in finding music[5]. Some of them e.g., [VP05, PALB+06], use several data sources and make use of metadata associated with the music. While this helps in supporting robustness, when considering a user's music collection, metadata may be not available, or its quality may not always be assured. For example, different sources of metadata may assign similar music into different categories. Based on such considerations, we assume that metadata is not available, but rather one music similarity measure based on the audio signal captures the similarity of the songs. The approach is based on an audio similarity measure that uses the well-known Mel Frequency Cepstral Coefficients (MFCCs) as a basis for similarity computation [AP04, ME05]. While such algorithms widely are evaluated by genre classification experiments, the one advantage of using them is that they do not assign a track to one (or more) classes, but rather one yields a usually continuous similarity score between each pair of songs. For example, by doing so, similar-sounding music that one usually would assign to different genres still can be considered as being similar. In [PPW05], a scenario is presented where similarities between all tracks on a user's player are known (or calculated from the audio data), and a playlist should be generated on the fly based on the user's current (dis)likes. To support mobile use, the user interface is very limited. Tracks to be played are selected based only based on the user's presses of a skip button. Several algorithms are suggested and compared, and the best performing seem to perform good enough for use in a real-world scenario. We adopt the ideas of these algorithms for the scenario presented here. Otherwise, the scenario described here is more complicated than the one in [PPW05], where each track is either played or skipped. Here, a particular track might be played

[5] An overview can be found at http://mirsystems.info/

and / or skipped multiple times. Furthermore, there is not only the decision which track should be played (or added to the player), but also which tracks the user does not like any more, and consequently should be removed from the player.

For evaluation of MIR techniques, generally the best approach is to conduct user evaluations (as e.g. in [VP05]) or surveys. As user studies are cost and time intensive, and our work is still in an experimental state, we follow the approach from [PPW05] and restrain to automated experiments where a certain user behavior is assumed.

3 Application Scenario

The basic usage scenario in the suggested application is as follows: When listening to music that is on her mobile player while being en route, the user skips those tracks she currently does not like. When returning home, she plugs the player to the personal computer. At this point, the application selects those tracks that should be removed from the player because the user does not like them so much any more. These tracks are transferred to the computer's hard disk. Also, the application selects those tracks available over the music flat rate that the user is likely to like. Those tracks are obtained from the service and transferred to the device. More formally, the scenario is as outlined in the following paragraphs.

3.1 Initial State

In the scenario of this paper, the user likes all tracks that are initially on the device. It is assumed that the user only adds music she likes to the device when she has just obtained it, which seems a quite reasonable assumption.

3.2 Listening Session

After the user has copied music she likes to the device, she uses the device to listen to music. It is assumed that during such a listening session the music on the device remains the same, i.e., there is no active connection to the music repository. This assumption is made for several reasons.

- The device itself may not have a connection (e.g., a rather simple stand-alone music player). If the device does have connectivity, then circumstances may prohibit its use (e.g., on airplanes connectivity may not be used). For these reasons, assuming that connectivity is not available makes the modeling more general.
- Assuming a device that is always connected to the music database (repository), the scenario is much closer to the one presented in [PPW05], because basically all tracks are available for streaming. Thus, the problem reduces to selecting a track that currently best fits. However, even in this scenario, it might be beneficial to know which tracks the user has disliked in the past and thus is likely to dislike also currently.

An important aspect of a listening session is how many tracks k_i the user actually listens to in listening session i before the session ends and the device is being updated. In the experiments presented here, this is a random number in the range from 15 to 25, corresponding to approximately 45 to 75 minutes of music. For better comparability between parameter settings, k_i is the same for each experiment (however, not the same for each i).

Another important factor is how the next track is selected that is played to the user. There are at least three possible algorithms:

1. The tracks are presented in a fixed predetermined order, e.g. in alphabetical order or in the order they were added to the device.
2. An algorithm based on skipping behavior is used during the listening session [PPW05].
3. Tracks are presented in random order.

If the track update algorithm works well, then most tracks on the device fit the user's taste and thus are acceptable for the user. However, when using fixed presentation order (case 1), the first tracks may become boring to the user, provoking the user to skip the first tracks, which adds false skips (i.e., skips that do not indicate a genuine dislike). Presenting tracks in random order (case 3) is an algorithm implemented in most current audio players, i.e., a de facto standard. This, and the fact that using an algorithm based on skipping behavior (case 2) might induce unwanted effects on the number of skips (e.g., such an algorithm may prefer or avoid certain tracks), makes it the algorithm of choice for the experiment.

3.3 User Feedback

There are several ways one could think of for the user to give feedback about her liking or disliking of the currently played music. The most simple user interface would consist only of a *skip* button the user presses if she dislikes the currently played track. Not pressing this button is interpreted as the user liking the played music.

However, in a real-world application, there might be a variety of reasons why the user presses the skip button. Besides accidental presses, most notably the user might press the button because she does not want to listen to this track right now (but in general she likes the track). Thus an algorithm based on such feedback data needs to be robust against such influences. To avoid ambiguities, it is possible to add a second button with which the user can indicate that she wants to *ban* the currently played track. In this scenario, pressing the *skip* button then has no particular meaning to the track discard algorithm. In the experiments to be described below, we simulated both skipping and banning.

3.4 User Modeling

For a systematic quantitative experimentation, we need to simulate a hypothetical user. Specifically, we need to define a criterion by which the user decides

whether or not she likes a piece or hits the skip/ban buttons. We assume two main use cases, comparable to [PPW05]:

- *Use Case 1:* The user only likes tracks from one genre. Generally, tracks from this genre are accepted, and tracks from all other genres are skipped.
- *Use Case 2:* In the beginning, the user only likes tracks from one genre (A). Over time, the user's preferences change to a different genre (B).

In the second case, tracks from genre A are accepted in the first $\frac{3}{4}$ of the listening sessions, and tracks from genre B are accepted during the last $\frac{3}{4}$ of the listening sessions.

Also we take into consideration that once the user has listened to a particular track many times, the track gets boring, and eventually the track is (almost) always skipped (cf. [CDB05]). In the use cases, this is formalized as a number $\texttt{maxListen}_t$ that is associated with each track t. Once the (hypothetical) user has accepted track t this often, she will never accept it again (i.e., always skip it afterwards). The value of $\texttt{maxListen}_t$ is chosen randomly in the range from 10 to 20 for each track and kept fixed for all different experiment settings.

Obviously, the algorithm is also influenced by the number of tracks that fit onto the device (denoted as `capacity`). Today, most mobile players have at least 1 GB of memory, which corresponds to about 140 tracks in high encoding quality (if a track has five minutes length and a bit rate of 192 kbit per second). A realistic alternative figure would be 4 GB (555 tracks). For the experiments, it is of importance to consider the relation of the number of tracks that fit on the device and the number of tracks that are available in the "repository" (i.e., the tracks available over the flat rate service). Particularly the latter is limited by the size of the music collection available for the experiments.

Finally, it is important to model unreliability of the user's input in the experiments: if the input is fully reliable, then pressing the skip button for a piece means that the skip button will always be pressed for this track (in the current experiment). Consequently, an algorithm can safely discard all tracks for which the skip button has been pressed once. As this is unrealistic, unreliability of the input is simulated by reversing the user's decision to press the skip button (or not to press it) with a certain probability. In our experiments, we use $p = 0.2$. The *ban* button, on the other hand, is always deterministically pressed when the simulated user does not like a piece (or does not like it any more), according to the present use case.

All these discussed parameters have to be considered when evaluating the algorithms. Lacking a basis, we set most of them in an ad-hoc manner, assuming that the relative performance of the evaluated algorithms with respect to each other will persist over a range of possible parameter settings. The evaluated algorithms are presented in the next section.

4 Evaluated Algorithms

As already indicated, the algorithm mainly consists of two parts: When the device is updated, first those tracks are selected that are not of interest to the user

any more. These tracks are removed from the device, and replaced by tracks selected in the second stage of the algorithm. Obviously, the first part of the algorithm (the *discard policy*) has a limiting effect on the potential benefit of the second part (the *track selection policy*). If the discard policy performs poorly, e.g., by removing too few outdated tracks from the device, then the track selection policy can not replace these tracks with better matching ones.

4.1 Discard Policy

For creating discard policies, it is of use to define a basic measure of acceptance called *like value* here. Based on the skip count s_t and the listen count l_t of track t, the like value of track t is

$$\mathtt{lv}_t = \frac{l_t}{l_t + s_t} \tag{1}$$

The like value is only defined for tracks that have already been presented to the user.

The following discard policies were evaluated. The first of them only assume the presence of a skip button, while the last is based on the ban button.

1. *skipped_once.* Discard *all* tracks that have been skipped. This is the algorithm that discards most tracks. Particularly, also tracks that have only accidentally been skipped are immediately discarded. This is formalized in the like value above: discard tracks with a like value lower than 1.0.
2. *skip_larger_accept.* Discard tracks that are more often skipped than accepted. Formally: discard tracks with a like value lower than 0.5. Obviously, this discard policy is much more conservative than policy 1.
3. *like_value.* As discard policy 2 might be too conservative, and 1 might discard too many tracks, in the third evaluated discard policy, a track is discarded if it has a like value lower than 0.75.
4. *userbanned.* The most informative discard policy is to discard exactly those tracks that are known to be not liked by the user (any more). This is the best possible policy, but comes at the cost of having an additional button on the device.

In a realistic setting, the user should be able to mark a track "sticky", so that it is not removed from the device. Here, this is omitted as such an option would further complicate the experimental setup.

4.2 Track Selection Policy

To decide which songs to add to the player, the system relies on an *audio similarity measure* that compares tracks on the mobile device to tracks in the repository. More specifically, we use the following similarity measure here: Each track is represented as a Gaussian model of Mel Frequency Cepstral Coefficients (MFCCs) calculated on short audio frames. Two songs are compared by calculating the KullbackLeibler (KL) distance between their models. More detailed descriptions of the approach can be found in the literature ([AP04, Pam06]).

Based on this, the following track selection policies were evaluated (cf. [PPW05]):

1. *Random Baseline.* Added tracks are selected from the "candidate" tracks in the repository in a random manner.
2. *Policy A.* The order in which new tracks are added to the device is determined only based on the tracks that are initially on the device. All tracks in the repository are ordered once by their minimum distance to any of the tracks initially on the device.
3. *Policy B.* Those songs closest to any of the songs listened to during the last listening session are added.
4. *Policy C.* Select songs closest to any of the last `capacity` tracks[6] that have been accepted (i.e., listened to).
5. *Policy D.* Select songs based on the last `capacity` tracks that have been accepted (set A) and the last `capacity` tracks that have been skipped (set R). Let d_a be the distance of song d to the closest song from A, and d_r the distance from song d to the closest song in R. From all tracks with $d_a < d_r$, select those with smallest d_a and transfer them to the player. If these are not enough tracks to fill the player, go on with adding those tracks with smallest $\frac{d_a}{d_r}$. Optionally, the distance calculation is weighted by the number each track in A and R was listened to or skipped, respectively.

To determine the distance of a candidate track in the repository to the tracks on the device (i.e., the subset used to determine the distance), the minimum distance of the track to all tracks of interest is used [Log04]. Policies B, C and D were also evaluated in a second variant, where distances to the tracks on the device were weighted according to the number of times the corresponding tracks have been accepted. Distances to tracks that have more frequently been listened to were weighted higher. In this case, the final distance is the weighted mean distance instead of the minimum.

For all policies the general rule applies that a song that already has been discarded from the device in the past will not be added again. Of course, in a real-world application, the user should be allowed to do so manually, which could eventually improve the quality of the suggested songs.

5 Evaluation Results

To measure the quality of an algorithm, two measures were used: the average number of times the skip button was pressed by the (simulated) user, and the number of tracks discarded by an algorithm. In both cases, generally lower means better. However, of course if too few tracks are discarded, then the number of times the skip button is pressed increases.

For *userbanned*, it is of interest how often skip or ban was pressed, both are counted together. The skip button presses are also counted to allow a comparison

[6] Or fewer, if fewer than `capacity` tracks have been accepted so far.

to the other policies, i.e., to also take into account the effects of the random factor p that is only applied to the skip button, but not to the ban button. This results in the reasonable situation where each button press is counted as a user interaction. For the other track discard policies besides *userbanned*, it is only of interest how often skip was pressed, as ban presses are ignored by the other policies. Some parameters used in the experiments are given in Table 1.

Table 1. Parameters used in the experiments

Parameter	Value
`capacity` (# of tracks on device)	100
p (randomly invert skip decision)	0.2
# listening sessions	100
k_i (# of tracks listened to in session i)	15 to 25 (randomly chosen for each i)
$\texttt{maxListen}_t$	10 to 20 (randomly chosen for track t)

5.1 Data

We evaluated the various approaches on a music collection consisting of 51,056 tracks by 7,610 artists. Tracks were assigned to 45 genres. For the experiments, the 18 largest genres were used (while all tracks were left in the collection). Small genres were disregarded because potentially they did not contain enough tracks to offer enough matches (e.g., in cases where the discard policy would discard very many tracks), which could have resulted in artificially low evaluation results. All used genres consisted of more than 1,500 tracks each. Altogether, the 18 largest genres contained 35,265 tracks. For evaluation, each of the 18 genres was assumed to be "accepted" music in turn, and results were averaged.

5.2 Use Case 1

Figure 1 gives the average number of button presses that sum up during the considered simulated usage time of the device (100 listening sessions) when only tracks by a particular genre are accepted. For better comparability, values are given relative to the best obtained value of the discard policy *userbanned*, which is thought to give an indication of an upper bound. On average over all policies, 1,933 out of a total of 3,909 suggested tracks were fully listened to by the hypothetic user during all listening sessions.

The results show that even for the better-performing (non-baseline) algorithms, overall in about 50% of the cases a suggested song was skipped. This seems a large number, but it should be considered that even for a perfect algorithm this would still be 20% because of the random factor. To evaluate if these results are low enough for a real-world application, a user study is necessary.

A closer look at the relative performance of the algorithms shows that as expected, the best discard policy is *userbanned*. The other discard policies produce much higher values, starting at a factor of 1.8 times as high.

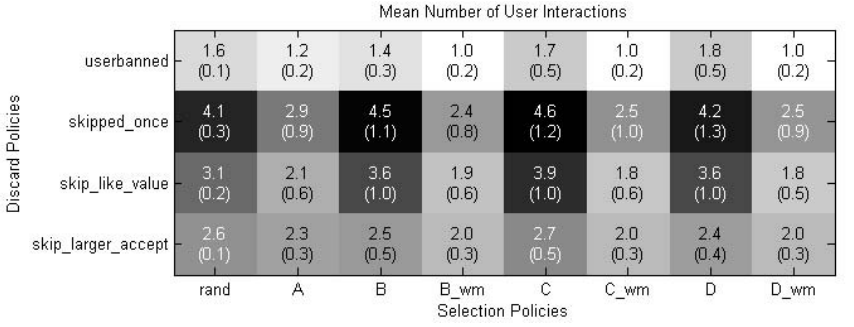


Fig. 1. UC1: Mean and standard deviation of number of *user interactions* (button presses), summed over 100 listening sessions. Mean and standard deviation measure this value over 18 genre experiments. Values are normalized with respect to the best (minimum) average value obtained for *userbanned*, which is 1058.5.

For the track selection policies, it is somewhat surprising that the simple *random* policy which was used as a baseline does not always perform worse than more sophisticated approaches (*B*, *C*, *D*). However, all of *B*, *C* and *D* are greatly improved when the similarity measure is weighted according to the number of times a song has actually been listened to (*B_wm*, *C_wm* and *D_wm*). In combination with *like_value*, they produce the best results achieved in the experiments in cases where only a skip button is available. Taking into account the number of times a song was accepted seems an important step, resulting in the best values. A likely reason is that the random factor is leveled out to a certain degree. Also it is interesting to note that all three algorithms that use weighted values perform quite similarly, which may be an indication that this is some sort of upper bound for the examined techniques.

To complement the number of button skips, Figure 2 gives the number of songs that were removed from the device (and replaced with new tracks) in the

Fig. 2. UC1: Mean and standard deviation of number of *discarded songs*, measured relative to the minimum mean number of user interactions for *userbanned*, which is 607.1.

same period. As expected, using the discard policy that discards tracks most reluctantly (*skip_larger_accept*) leads to the lowest number of discarded tracks. These numbers are even lower than for *userbanned*. Also, it is of interest that the number of discarded songs decays from *skipped_once* over *skip_like_value* to *skip_larger_accept*, while the number of user interaction was lowest for *skip_like_value*, which we see as an indication that this is the most recommendable of the presented *skip* policies.

5.3 Use Case 2

Figures 3 and 4 give the corresponding data based on use case 2, when the user's preferences change over time (i.e., the genre accepted by the user shifts from a genre A to genre B). Each of the 18 genres was used as the starting genre A in turn. A corresponding genre B was chosen manually, where each genre appeared once as genre B. Most transitions were between somewhat related genres, although there also was the unusual transition *Classical* to *Pop-Rock*. Generally, the results show the same tendency as for use case 1. Again, the

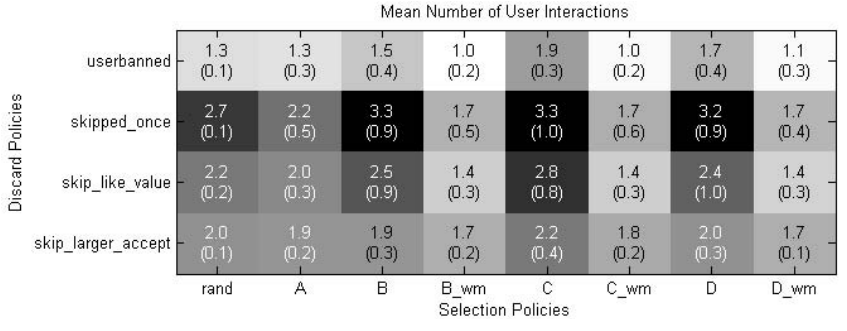


Fig. 3. UC2: Mean and standard deviations of number of *user interactions* (button presses), measured relative to the minimum mean number of user interactions for *userbanned*, which is 1383.

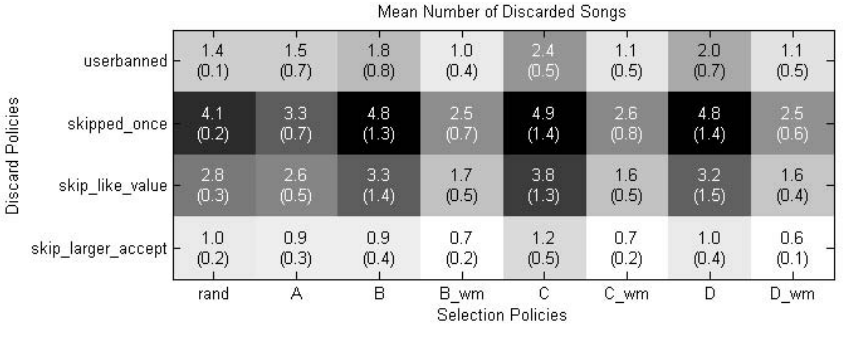


Fig. 4. UC2: Mean and standard deviations of number of *discarded songs*, measured relative to the minimum mean number of discarded songs for *userbanned*, which is 928.5

combination of track discard policy *skip_like_value* with selection policy *D_wm* resulted in the lowest values in the non-baseline scenario where only a skip button is available, although there is no big difference to the other two algorithms based on weighted values.

6 Conclusion

We have presented several algorithms aiming to automatically select music that is on a user's portable music player. The algorithms assess the user's like and dislike of individual tracks by the number they were skipped. Tracks the user seems to dislike are removed from the device, and replaced by tracks that seem to fit the user's current music taste. Our automated experiments showed clear differences between the suggested algorithms, while weighting tracks based on the number they were skipped improves the examined algorithms to a comparable level with regard to the evaluation measures. User studies could show if the algorithms are sufficiently good for using them in a real world application, and to measure the actual user experience.

Acknowledgments

This work is supported by the Austrian *Fonds zur Förderung der Wissenschaftlichen Forschung (FWF)* under grants L112-N04 and L511-N15.

References

[AP04] Aucouturier, J.-J., Pachet, F.: Improving timbre similarity: How high is the sky? Journal of Negative Results in Speech and Audio Sciences 1(1) (2004)

[CDB05] Cunningham, S.J., Downie, J.S., Bainbridge, D.: "The Pain, the Pain": Modelling Music Information Behavior and the Songs we Hate. In: Proceedings of the 6th International Conference on Music Information Retrieval, ISMIR 2005 (2005)

[Log04] Logan, B.: Music Recommendation From Song Sets. In: Proceedings of the 5th International Conference on Music Information Retrieval, ISMIR 2004 (2004)

[ME05] Mandel, M., Ellis, D.: Song-Level Features and Support Vector Machines for Music Classification. In: Proceedings of the 6th International Conference on Music Information Retrieval, ISMIR 2005 (2005)

[PALB+06] Pachet, F., Aucouturier, J.-J., La Burthe, A., Zils, A., Beurive, A.: The cuidado music browser: An end-to-end electronic music distribution system. Multimedia Tools and Applications 33(1), 331–349 (2006); Special Issue on the CBMI 2003 Conference

[Pam06] Pampalk, E.: Computational Models of Music Similarity and their Application in Music Information Retrieval. Docteral dissertation, Vienna University of Technology, Austria (March 2006)

[PPW05] Pampalk, E., Pohle, T., Widmer, G.: Dynamic Playlist Generation Based on Skipping Behaviour. In: Proceedings of the 6th International Conference on Music Information Retrieval, ISMIR 2005 (2005)

[VP05] Vignoli, F., Pauws, S.: A Music Retrieval System Based on User-Driven Similarity and its Evaluation. In: Proceedings of the 6th International Conference on Music Information Retrieval, ISMIR 2005 (2005)

Music Thumbnailing Incorporating Harmony- and Rhythm Structure

Björn Schuller, Florian Dibiasi, Florian Eyben, and Gerhard Rigoll

Institute for Human-Machine Communication, Technische Universität München,
Arcisstrasse 21, 80333 München, Germany
schuller@tum.de
http://www.mmk.ei.tum.de

Abstract. A variety of approaches exist to the automatic retrieval of the key part within a musical piece - its thumbnail. Most of these however do not use adequate modeling with respect to either harmony or rhythm. In this work we therefore introduce thumbnailing that aims at adequate musical feature modeling. The rhythmic structure is extracted to obtain a segmentation based on beats and bars by an IIR comb-filter bank. Further, we extract chroma energy distribution normalized statistics features of the segmented song improving performance with dB(A) and pitch correction. Harmonic similarities are determined by construction and analysis of a similarity matrix based on the normalized scalar product of the feature vectors. Last, thumbnails are found lending techniques from image processing. Extensive test runs on roughly 24 h of music reveal the high effectiveness of our approach.

1 Introduction

A wide variety of applications uses or potentially benefits from music thumbnails in order to provide an insight into songs such as pre-hear functions in online music stores, teaser previews on the radio, samples for deejays to create mega-mixes or efficiently browsing and arranging (cf. e.g. [Stober and Nürnberger, 2008]) through large music collections, e.g. on a mobile MP3 player. In addition query by example systems (e.g. [Schuller et al., 2003]) highly benefit from pre-extracted thumbnails to build up the required database for similarity matching with sung or hummed queries. Nowadays these thumbnails usually have to be generated manually due to the lack of appropriate and robust methods which are capable to accomplish reliable automatic generation. Especially for acoustic formats which deal with real audio (we use the popular MPEG-1 Audio Layer 3 standard) there are no known prosperous methods or systems so far.

Since voiced, repeating sequences such as chorus sections are believed to be the most mnemonic parts of songs [Burges et al., 2004], the approach described in this work aims at extracting the chorus by successfully combining different ideas of previous works. There are several works dealing with extracting audio thumbnails or determining the musical structure of songs. They can be divided

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 78–88, 2010.

into two general types regarding the feature nature they are based upon. Alternatively they can also be classified according to their kind of approach, such as building up and analyzing similarity matrices or applying a segmentation step with a subsequent clustering or classification.

Burges et al. [Burges et al., 2004] create their own features by applying a modulated complex lapped transform followed by a logarithmizing step and by reducing the information using oriented principal component analysis. A clustering algorithm is applied to merge similar sequences which are then classified by using a scaled Renyi entropie and spectral flatness. Logan et al. [Logan and Chu, 2000] extract Mel frequency cepstral coefficients (MFCC) and cluster the song using a modified Kullback Leibler distance. The second approach presented in [Logan and Chu, 2000] models each song by an ergodic hidden Markov model (HMM) in order to extract the musical structure. Aucouturier et al. use ergodic HMM with 3 states by approximating the spectral envelope with MFCC, linear predictive coefficients and discrete cepstrum coefficients [Aucouturier et al., 2005] and with Gaussian mixture models initialized by a clustering step [Aucouturier and Sandler, 2001]. Foote et al. introduce the concept of music visualization by constructing a similarity matrix based on MFCC using the scalar product [Foote, 1999] and a normalized scalar product [Cooper and Foote, 2002]. Peeters et al. [Peeters et al., 2002] use dynamic features which maximize the trans-information and build up a similarity matrix. A segmentation step splits the song into small segments which are then processed with a clustering algorithm in order to generate an initialized set for ergodic HMM. Jehan [Jehan, 2005] first divides the music signal into several segments using an event detection function and then applies dynamic time warping to extract the musical structure.

Abdallah et al. [Abdallah et al., 2005] use an unsupervised Bayesian clustering model to extract musical structure by estimating its parameters using a modified Expectation Maximization algorithm. Bartsch et al. [Bartsch and Wakefield, 2001] perform a beat synchronous segmentation using a beat tracker followed by a similarity matrix based on chroma features. By applying uniform moving average filtering a time-lag matrix is obtained whose maximum element is located according to constraints regarding the minimum lag and the maximum occurrence of a section. Goto [Goto, 2006] extends this work by allowing modulated repetitions and by using an adapted measure to select chorus sections. Müller et al. [Müller and Kurth, 2006] present an approach for enhancing similarity matrices by introducing a new set of features based on harmonic information.

From these works we learn that beat-synchronous feature extraction is advantageous provided a reliable detection. Features that respect the musical background of the signal, such as chroma, are superior to e.g. MFCC, and it seems favorable to incorporate temporal information as in [Müller and Kurth, 2006]. Finally, retrieving the pre-dominant parts by similarity matrices seems most promising, and dynamic modeling, such as Dynamic Time Warp (DTW), is usually rather contra-productive provided beat-synchrony. In the next section we

describe a system built upon these considerations that lends simple but fast techniques from image processing for the similarity matching and enhances features by perceptive modeling. We also define measures for evaluation and report results on a full day of MP3 compressed original audio recordings from diverse genres.

2 Feature Extraction

2.1 Rhythm Information

To extract rhythmic structure the highly robust beat tracker introduced in [Schuller et al., 2007], [Eyben et al., 2007], which bases on fundamentals introduced in [Scheirer, 1998], is used in the ongoing. After a preprocessing step which involves down-sampling to 11 025 kHz and transforming into the frequency domain, the signal is filtered with the A-weighting function according to the human perception of sound. In order to reduce the number of bands without loosing rhythmic information the audio signal is split into frequency bands using a bank of 12 overlapping triangular filters which are equidistant on the Mel-Frequency scale.

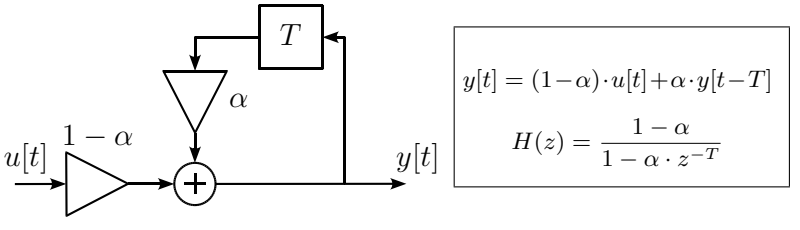


Fig. 1. Block diagram (left), difference equation (top) and transfer function (bottom) of an IIR comb filter

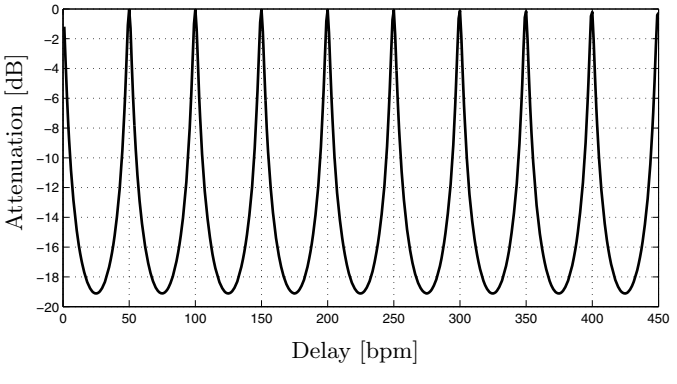


Fig. 2. Magnitude response for an IIR comb filter with gain $\alpha = 0.8$ and base tempo $50bpm$

Next, the envelope of each band is extracted using a half wave raised cosine filter and weighted by incorporating the moving average over the previous 10 and the following 20 samples. This is done due to the fact that humans perceive note onsets louder if they occur after a longer time of lower sound level. Hence, we determine the lowest metrical level referred to as tatum grid using a bank of 57 phase comb filters with gain $\alpha = 0.8$ and delays ranging from $\tau = 18$ to $\tau = 74$ envelope samples. A comb filter is able to extract a frequency and its multiples by adding a delayed version of the signal to itself. This filter is specified by the gain α and the delay τ. An example for such a comb filter is depicted in figure 1, its magnitude response for $\alpha = 0.8$ and $\tau = 50$ bpm is illustrated in figure 2. Based on the tatum grid our beat tracker is able to determine meter and tempo features by setting up narrow comb filter banks centered on multiple tempos of the tatum grid.

2.2 Harmonic Information

In order to incorporate the temporal harmonic structure of a song we use the chroma energy distribution normalized statistics introduced by Müller et al.

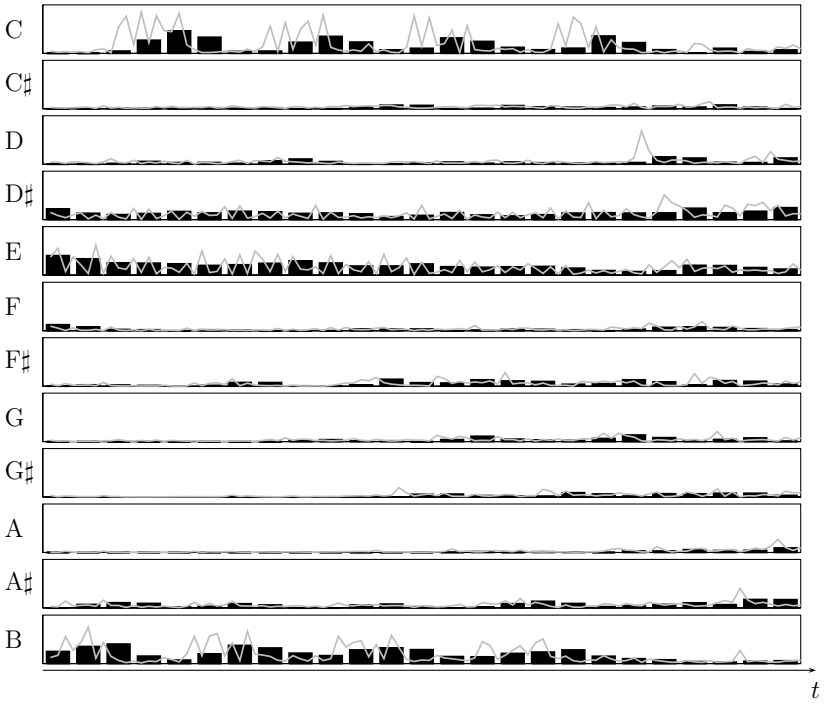


Fig. 3. Harmonic representation of the first 20 seconds of Abba - Mamma Mia. The light curves illustrate the local chroma energy distribution, the dark bars the CENS features.

Müller et al., 2005. These features are based on chroma features which are computed using a fast Fourier transform with a window length of 372 ms and an overlap of 0.5 by taking into account a psychoacoustic model using A-weighting filtering as within the beat tracking according to DIN EN 61672-1:2003-10 and by decomposing the audio signal into frequency bands representing the semitones which are defined for equal temperament as

$$f_i = f_0 \cdot 2^{i/12} \qquad f_0 = f(\mathrm{A0}) = 27.5\,\mathrm{Hz} \tag{1}$$

with $15 \leq i \leq 110$ (corresponding to the notes C2–B9) and therefore covering 96 semitones (8 octaves). In order to overcome deficient recordings due to mis-arranged recording settings or intentional manipulations of the sound impression, pitch correction is applied. A long term frequency analysis computes the prominent frequency f_p and determines a factor c

$$c = \frac{f_p}{f_r} \tag{2}$$

with

$$f_r = \underset{f_i}{\operatorname{argmin}} \left\| \frac{f_p}{f_i} - 1 \right\| \tag{3}$$

Next, all semitones f_i are multiplied with the factor c to correct their pitch. In order to allocate the frequencies to the semitones a nearest neighbor approach is applied which implies the use of Gaussian bells $g_i(x)$ centered at f_i given by

$$g_i(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{\left(\frac{x-f_i}{f_i-f_{i-1}}\right)^2}{2\sigma^2}} \qquad \sigma = 0.125 \tag{4}$$

Now we normalize the resulting sub-bands s_i by dividing each one belonging to the same octave O by the sum of these sub-bands according to

$$\hat{s}_i = \frac{s_{i,O}}{\sum s_{i,O}} \qquad s_{i,O} = s_i \in O \tag{5}$$

In a final step we add up all sub-bands corresponding to the same relative pitch class, for example for the chroma C we compute $\overline{s}_1 = \hat{s}_{15} + \hat{s}_{27} + \ldots + \hat{s}_{99}$, and normalize the resulting values

$$v_i = \frac{\overline{s}_i}{\sum \overline{s}_i} \qquad 1 \leq i \leq 12 \tag{6}$$

Due to the fact that the local chroma features are too sensitive concerning articulation effects and local tempo deviations we extend the chroma features following Müller et al. Müller et al., 2005 and apply to each component of $\mathbf{v} = (v_1, \ldots, v_{12})$ a quantization function Q defined as

$$Q(a) := \begin{cases} 4 & \text{for} \quad 0.4 \leq a \leq 1 \\ 3 & \text{for} \quad 0.2 \leq a < 0.4 \\ 2 & \text{for} \quad 0.1 \leq a < 0.2 \\ 1 & \text{for} \quad 0.05 \leq a < 0.1 \\ 0 & \text{for} \quad 0 \leq a < 0.05 \end{cases} \tag{7}$$

In the next step, we convolve 11 consecutive quantized chroma vectors $Q(\mathbf{v}(i))$ component-wisely using a Hann window which results in a weighted 12-dimensional features vector including temporal harmonic information. As the signal change due to the windowing is quite slow, down-sampling with a factor of 4 is applied. The resulting feature vectors are referred to as chroma energy distribution normalized statistics (CENS) which we will denote from now on as $\mathbf{v} = (v_1, \ldots, v_{12})$. A comparison between the two types of features is visualized in figure 3.

3 Chorus Extraction

3.1 Similarity Matrix

In line with [Cooper and Foote, 2002] we compute a $N \times N$ similarity matrix $\mathbf{S}$ based on the normalized scalar product

$$\mathbf{S}(i,j) = \frac{\langle \mathbf{v}(i), \mathbf{v}(j) \rangle}{\|\mathbf{v}(i)\| \cdot \|\mathbf{v}(j)\|} \tag{8}$$

In terms of music visualization we are looking for bright diagonals (we mean diagonal segments parallel to the main diagonal, c.f. figure 4) in matrix $\mathbf{S}$ which correspond to similar segments in a song. Thus, we use an edge filter given by

$$F_{Diag}(i,j) = \begin{cases} 1 & \text{for} \quad i = j \\ c & \text{for} \quad 0 < |i-j| \leq b \\ 0 & \text{for} \quad |i-j| > b \end{cases} \tag{9}$$

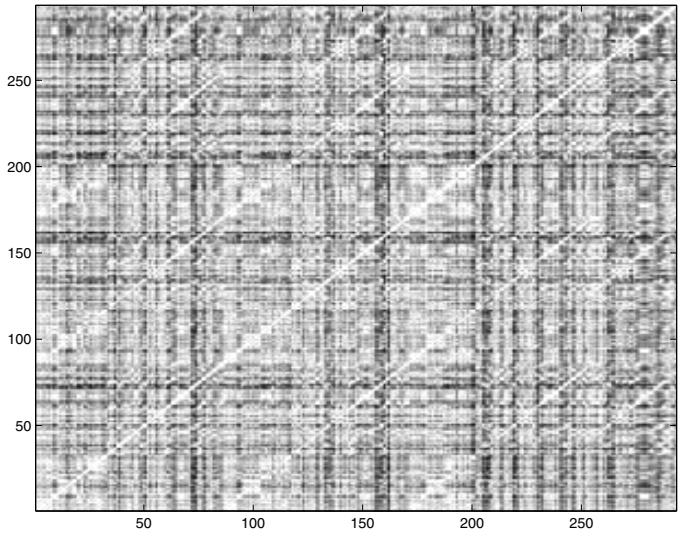


Fig. 4. Similarity matrix for Adriano Celentano - Azzurro. Bright diagonals illustrate a high similarity between two segments.

with $1 \leq i, j \leq 20$, $b = 5$ and $c = -\frac{2}{17}$, to extract these bright diagonals from the similarity matrix. After a normalization step a threshold δ is subtracted from the filtered image resulting in the matrix $\hat{\mathbf{S}}$ in order to reduce noise that is generated by the edge filtering. δ corresponds to the highest value which is exceeded by at least $10 \cdot N$ values of the filtered image. In a subsequent step, we create a binary matrix $\mathbf{S}_b$ according to

$$\mathbf{S}_b(i,j) = \begin{cases} 1 & \text{for} \quad \hat{\mathbf{S}}(i,j) > 0 \\ 0 & \text{for} \quad \hat{\mathbf{S}}(i,j) \leq 0 \end{cases} \tag{10}$$

3.2 Regions of Interest

Now we determine regions of interest (ROI). Starting and ending points of potential chorus sections are extracted by the following approach: Let $d(i,j)$ denote the temporal derivate along a diagonal segment $\mathbf{S}_b(i+1, j+1) - \mathbf{S}_b(i,j)$. In a first step segment bounds are estimated by setting starting points at i and at j if $d(i,j) > 0$ and corresponding ending points at i and j if $d(i,j) < 0$. In order to correct these preliminary bounds we introduce a counter c_k for each segment

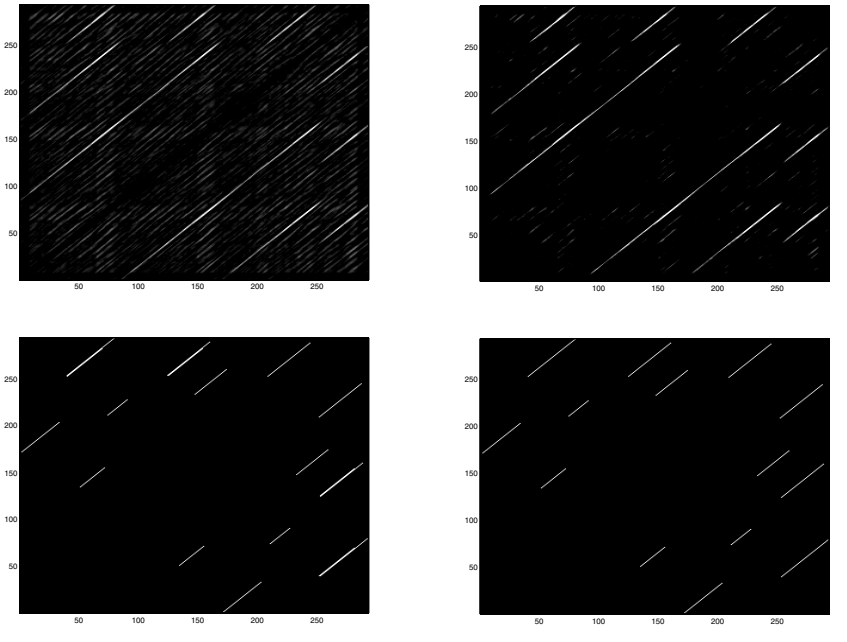


Fig. 5. Similarity matrices after the processing steps: first, edge filtered (top left), then dynamic thresholding (top right). From the resulting matrix $\hat{\mathbf{S}}$, respectively its binary representation $\mathbf{S}_b$, regions of interest are determined by length and characteristics of each segment (bottom left). Last, adjacent segments are combined (bottom right).

k and define a threshold δ_{Sim} which corresponds to the highest value exceeded by at least $0.1 \cdot N^2$ entries of the matrix $\mathbf{S}$. Starting at the middle (x, y) of each segment, we increment c_k, if $\mathbf{S}(x, y)$ falls below δ_{Sim}, and decrement c_k if it exceeds δ_{Sim} up to a minimal counter value of $c_k = 0$. If c_k is smaller than C we process the next value $(x, y) := (x - 1, y - 1)$. Otherwise we stop and save the corrected starting point $(x + C, y + C)$. Next, we apply these steps for the other directions with $(x, y) := (x + 1, y + 1)$ starting again at the middle of segment k gaining the corrected ending point $(x - C, y - C)$. The algorithm with $C = 4$ has delivered the best performance in practice. The described process is depicted by an example in figure 5.

In order to reduce the amount of regions of interest we define lower and upper limits for the segment length l. A dynamic lower bound given by

$$l \cdot m_{Sim} > 8.7\,\mathrm{s} \tag{11}$$

where m_{Sim} is denoting the mean similarity of the segment in the similarity matrix $\mathbf{S}$. This has proven as an optimal choice to eliminate short repeating sections containing non-relevant segments. Further, a static upper bound given by 29.1 s has shown good results to distinguish between chorus sections and longer sections such as verse or verse plus chorus. In a last step we combine adjacent segments as they do not contain any additional information.

We now define an audio thumbnail by taking the best remaining segment regarding its mean similarity m_{Sim}, as we assume chorus sections to be the most similar sequences among the regions of interest. In order to evaluate the regions of interest and to provide multiple thumbnails for each song, we extract the 3 best segments. These are optionally aligned to the automatically extracted beat boundaries and, if not too distant, the on-beat, to start at a well defined position.

4 Results and Discussion

Our approach was tested on a database containing 360 songs of different genres with an overall duration of 23 h 47 min. The database consists of 110 songs belonging mainly to Rock and Oldies on the one hand, and of 250 songs indexed by genre each with 50 songs covering Electronic Dance, Pop, Rock, German Folk, and Oldies on the other hand.

The evaluation was performed by comparing the initial positions of the extracted thumbnails to those manually annotated. If the deviation between these

Table 1. Correctly extracted audio thumbnails for different maximal deviations

T_{max}[s]	Top1 [%]	Top2 [%]	Top3 [%]
1	22.6	37.8	45.8
2	48.6	67.2	73.3
3	60.6	76.1	81.4

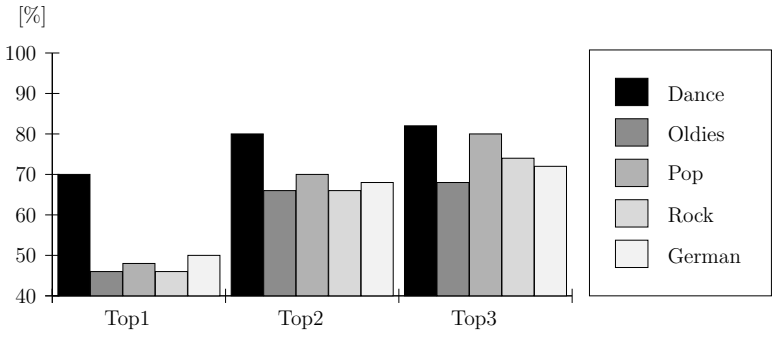


Fig. 6. Correctly extracted audio thumbnails depending on the evaluated genres for $T_{max} = 2\,\mathrm{s}$

positions was at most T_{max} the extracted audio thumbnail was assumed correct. The notation Top$X$ represents the percentage of the songs where one of the X best thumbnails was correctly extracted. Table 1 shows the results for a maximal allowed deviation of $T_{max} = 1, 2$ and 3 s. Most of the not correctly extracted thumbnails represent a characteristic part of the song as well, such as the chorus section with a deviation slightly above T_{max}, a non-equivalent repetition of the chorus or the beginning of verse or bridge. Unfortunately, no other work provides quantitative results of the generated audio thumbnails in terms of deviation from the actual chorus sections. Therefore, no objective comparison is possible at this time.

Figure 6 illustrates the results specific to the genre. The notably higher performance for electronic dance results from the fact that electronic music provides higher similarities due to perfect accordance of the electronically produced tones. Further, the musical structure is generally simpler and fewer variations are found.

5 Conclusion and Outlook

Within this paper we presented an effective approach for automatic extraction of audio thumbnails based on rhythmic structure and harmonic similarity analysis. Experimental test runs provided promising results, especially for electronic dance music where we were able to determine the chorus section for 70 % of the songs with a maximal deviation of $\pm 2\,\mathrm{s}$. Likewise, we could "compress" one day of music to roughly half an hour of thumbnails.

In future works the algorithm can easily be extended by additional modules to increase the performance by incorporating key changes (cf. e.g. [Chew, 2002]), progression structure (cf. e.g. [Lee and Slaney, 2008, Schuller et al., 2008], or by classifying vocal and non-vocal sequences (cf. e.g. [Berenzweig and Ellis, 2001]). Also, a variety of further matching algorithms can be comparatively considered (cf. e.g. [D'Aguanno and Vercellesi, 2008]).

References

[Abdallah et al., 2005] Abdallah, S.A., Noland, K., Sandler, M., Casey, M., Rhodes, C.: Theory and Evaluation of a Bayesian Music Structure Extractor. In: Proc. 6th ISMIR, pp. 420–425 (2005)

[Aucouturier et al., 2005] Aucouturier, J.-J., Pachet, F., Sandler, M.: "The way it Sounds": Timbre Models for Analysis and Retrieval of Music Signals. IEEE Transactions on Multimedia 7(6), 1028–1035 (2005)

[Aucouturier and Sandler, 2001] Aucouturier, J.-J., Sandler, M.: Segmentation of Musical Signals Using Hidden Markov Models. In: Proc. of the Audio Engineering Society 110th Convention (2001)

[Bartsch and Wakefield, 2001] Bartsch, M.A., Wakefield, G.H.: To Catch a Chorus: using Chroma-Based Representations for Audiothumbnailing. In: Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, pp. 15–18 (2001)

[Berenzweig and Ellis, 2001] Berenzweig, A., Ellis, D.: Locating singing voice segments within musical signals. In: Proc. Int. Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), Mohonk, NY, pp. 119–123 (2001)

[Burges et al., 2004] Burges, C.J.C., Plastina, D., Platt, J.C., Renshaw, E., Malvar, H.S.: Duplicate Detection and Audio Thumbnails with Audio Fingerprinting. Technical Report MSR-TR-2004-19, Microsoft Research, MSR (2004)

[Chew, 2002] Chew, E.: An algorithm for determining key boundaries. In: Proc. 2nd Int. Conf. on Music and Artificial Intelligence (2002)

[Cooper and Foote, 2002] Cooper, M., Foote, J.: Automatic Music Summarization via Similarity Analysis. In: Proc. 3rd ISMIR, pp. 81–85 (2002)

[D'Aguanno and Vercellesi, 2008] D'Aguanno, A., Vercellesi, G.: Automatic synchronization between audio and partial music score representation. In: Proc. 6th Workshop on Adaptive Multimedia Retrieval (AMR 2008), Berlin, Germany (2008)

[Eyben et al., 2007] Eyben, F., Schuller, B., Reiter, S., Rigoll, G.: Wearable Assistance for the Ballroom-Dance Hobbyist - Holistic Rhythm Analysis and Dance-Style Classification. In: Proc. ICME, pp. 92–95 (2007)

[Foote, 1999] Foote, J.: Visualizing Music and Audio using Self-Similarity. In: Proc. 7th ACM Int. Conf. on Multimedia (Part 1), pp. 77–80 (1999)

[Goto, 2006] Goto, M.: A Chorus Section Detection Method for Musical Audio Signals and its Application to a Music Listening Station. IEEE Transactions on Audio, Speech and Language Processing 14(5), 1783–1794 (2006)

[Jehan, 2005] Jehan, T.: Hierarchical Multi-Class Self Similarities. In: Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, pp. 311–314 (2005)

[Lee and Slaney, 2008] Lee, K., Slaney, M.: Acoustic chord transcription and key extraction from audio using key-dependent hmms trained on synthesized audio. IEEE Transactions on Audio, Speech and Language Processing 16, 291–301 (2008)

[Logan and Chu, 2000] Logan, B., Chu, S.: Music Summarization Using Key Phrases. In: Proc. ICASSP, vol. 2, pp. 749–752 (2000)

[Müller and Kurth, 2006] Müller, M., Kurth, F.: Enhancing Similarity Matrices for Music Audio Analysis. In: Proc. ICASSP, vol. 5, pp. 9–12 (2006)

[Müller et al., 2005] Müller, M., Kurth, F., Clausen, M.: Chroma-Based Statistical Audio Features for Audio Matching. In: Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, pp. 275–278 (2005)

[Peeters et al., 2002] Peeters, G., Burthe, A.L., Rodet, X.: Toward automatic music audio summary generation from signal analysis. In: Proc. 3rd ISMIR, pp. 94–100 (2002)

[Scheirer, 1998] Scheirer, E.: Tempo and beat analysis of acoustic musical signals. Journal of the Acoustic Society of America 103(1), 588–601 (1998)

[Schuller et al., 2007] Schuller, B., Eyben, F., Rigoll, G.: Fast and Robust Meter and Tempo Recognition for the Automatic Discrimination of Ballroom Dance Styles. In: Proc. ICASSP, vol. 1, pp. 217–220 (2007)

[Schuller et al., 2008] Schuller, B., Eyben, F., Rigoll, G.: Beat-synchronous data-driven automatic chord labeling. In: Proc. DAGA 2008, Dresden, Germany, pp. 555–556. DEGA (2008)

[Schuller et al., 2003] Schuller, B., Zobl, M., Rigoll, G., Lang, M.: A Hybrid Music Retrieval System using Belief Networks to Integrate Queries and Contextual Knowledge. In: Proc. ICME, vol. 1, pp. 57–60 (2003)

[Stober and Nürnberger, 2008] Stober, S., Nürnberger, A.: Towards user-adaptive structuring and organization of music collections. In: Proc. 6th Workshop on Adaptive Multimedia Retrieval (AMR 2008), Berlin, Germany (2008)

Automatic Reduction of MIDI Files Preserving Relevant Musical Content

Søren Tjagvad Madsen[1,2], Rainer Typke[2], and Gerhard Widmer[1,2]

[1] Department of Computational Perception, Johannes Kepler University, Linz
[2] Austrian Research Institute for Artificial Intelligence (OFAI), Vienna
soren.madsen@ofai.at, rainer.typke@ofai.at, gerhard.widmer@jku.at

Abstract. Retrieving music from large digital databases is a demanding computational task. The cost of indexing and searching depends on the computational effort of measuring musical similarity, but also heavily on the number and sizes of files in the database. One way to speed up music retrieval is to reduce the search space by removing redundant and uninteresting material in the database. We propose a simple measure of 'interestingness' based on music complexity, and present a reduction algorithm for MIDI files based on this measure. It is evaluated by comparing reduction ratios and the correctness of retrieval results for a query by humming task before and after applying the reduction.

1 Introduction

The brute force approach of searching everything in a large digital database is a demanding computational task. For music retrieval, we propose to preprocess each item in the database by removing uninteresting material in an intelligent way inspired by results from music cognition. As music collections can be very large, simplifying the data will allow us to search more files within the time/space limitations.

We introduce a number of reduction mechanisms and evaluate them by comparing storage requirements and the correctness of retrieval results for a music query task before and after applying the reduction. Initial results indicate that the approach provides a good tradeoff between the correctness of the retrieved files and the amount of additional data that can potentially be searched.

The paper is organised as follows. Section 2 describes the music search engine. Section 3 introduces the conceptual background of the reduction algorithm which is presented in section 4. Finally we report our findings in section 5.

2 A Melody Search Engine

We test our reduction method with the aid of a melody search engine and a small dataset of 1000 MIDI files. The MIDI search engine of Musipedia.org was applied

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 89–99, 2010.

to a set of MIDI files that were also used for the 'symbolic melodic similarity' task of MIREX 2006[1].

The web-based Musipedia searches about 100.000 mostly polyphonic MIDI files for melodic user queries, and the search engine uses the best performing algorithm for melodic similarity at MIREX 2006.

The goal is to reduce the material to be searched in each file, but at the same time having the search engine retrieve the same files before and after reducing. If the search engine retrieves the same files when querying the unprocessed and the reduced files, only uninteresting material has been removed.

2.1 The Search Task

Given a short user query (about 5 to 25 notes), the search engine will return the MIDI files that contain a sequence of notes which are melodically similar, that is, form a similar pattern of onset times and pitches. This is done in a transposition-invariant and tempo-invariant way. Results are ordered by similarity of the matching melody according to the Earth Mover's Distance [14]. Before queries can be executed, the search engine builds an index of all material in the database. This computationally expensive task is performed only once, and the index can then rapidly be queried.

2.2 Extracting Monophonic Melodies from Polyphonic Voices

As a preparation for finding melodies, the polyphonic MIDI files are first split into monophonic voices by using a skyline algorithm for every channel or track. Obtaining monophonic representations of polyphonic tracks and channels is essential for being able to measure the musical similarity, as the music similarity measure in the search engine is based on melodic similarity. Music similarity for arbitrary polyphonic constructs is much harder to define than music similarity between sequences of notes.

MIDI files with multiple tracks are split into single-track MIDI files (one for each track), and MIDI files with multiple channels into single channel MIDI files. For every single-track or single-channel MIDI file, a skyline algorithm inspired by [15] and [1] sorts the notes first by pitch (descending) and then by onset time (ascending). Then, starting with the highest pitch, the note with the earliest onset time – let us call it n – is marked as 'not to be deleted'. Every note with an equal or lower pitch and an onset time that is either the same or lies within 60 % of the duration of note n after n's onset is then deleted. From n, the algorithm proceeds to the next still existing and still unmarked note with the same pitch and treats it in the same way as n. Once all notes with n's pitch have been processed, the algorithm continues one half-tone lower, or it terminates in case there is no other unmarked note left. In the end, the remaining notes will form a sequence of the highest pitched notes in the voice.

[1] The Music Information Retrieval Evaluation eXchange, `http://www.music-ir.org/mirex/2006/index.php/Main_Page`

The musically rather uninformed skyline algorithm seems to be surprisingly good at agreeing with humans about what the melodic line is in a polyphonic set of notes [15,4] – probably because the melody in many cases is found as the top voice in a music arrangement. Nooijer [4] examined voices extracted with voice separation algorithms and voices extracted with skyline algorithms from eight MIDI files. Human listeners were asked to rank the voices in order to find the one best representing the overall melody of the piece. The skyline algorithm turned out to deliver the voice preferred by the evaluators in as many cases as the best, and more advanced, voice extraction algorithm.

2.3 Problem: Identifying Melodic Musical Material

The skyline algorithm extracts a good monophonic representation of a polyphonic voice, but it does not tell us whether there is anything melodic at all about the extracted sequence of notes. The whole track or channel on which the algorithm operates might contain only accompaniment figures which are very unlikely to ever be searched for by human users who would like to retrieve a MIDI file containing a given melody. Adding such potentially uninteresting material to the search engine's index makes the index unnecessarily large and slows down the search process.

If one could identify those notes which human users will probably search for, and then only index those notes, one should be able to work with a smaller index and potentially be able to include more files within the same space. But the melody search engine should still be able to give almost the same search results, so a conservative approach would be preferred. In the next sections, we describe and evaluate an algorithm which does exactly this – it reduces the size of MIDI files by excluding musical material which is unlikely to be perceived as relevant by humans.

3 Music Complexity as a Measure of Interestingness

The basic motivation for our model of melody identification is the observation, which has been made many times in the literature on music cognition, that there seems to be a connection between the complexity of a musical line, and the amount of attention that will be devoted to it on the part of a listener [13]. A voice introducing new or surprising musical material will potentially attract the listener's attention. This effect will, however, decrease if the new material is constantly repeated. We will pay less and less attention to it and become habituated or accustomed to the stimulus.

We postulate that in many situations this phenomenon of interestingness can be modeled by quantifying the amount of information that a given note sequence contains. If we assume that the musical foreground is the musical material that presents most new information and commands most attention, it seems natural to investigate music complexity as a method for detecting such material.

The idea of using information-theoretic complexity measures to characterise aspects of musical development is not at all new. After the advances in information theory in the forties, Leonard Meyer [10] was one of the first to relate music to information dynamics. He attempted to explain emotions and meaning in music in by examining the communication of uncertainties and expectations in music. Complexity has also been shown to be related to liking [11] in such a way that we tend to like the music the most that have medium complexity – too simple and too complex music is less preferred. Also, music complexity has been used for clustering music in genres [2,12,6], and in [5], a measure of Information Rate computed over a piece of music was shown to correlate in significant ways with familiarity ratings and emotional force response profiles by human subjects.

3.1 Measuring Musical Complexity

Earlier experiments of ours have shown that entropy of note events can with some success be used to point out melody notes and melody tracks in MIDI files [9,8]. Shannon's entropy is a measure of randomness or uncertainty in a signal. If the predictability is high, the entropy is low, and vice versa. Let X be a discrete random variable on a finite set $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ with probability distribution $p(x) = Pr(X = x)$. Then the entropy $H(X)$ of X is defined as:

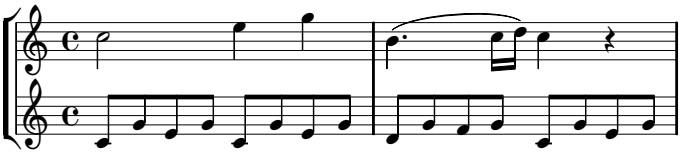
$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log_2 p(x). \tag{1}$$

X could for example be the set of MIDI pitch numbers in a note sequence and $p(x)$ would then be the probability (estimated by the frequency) of a certain pitch. In the case that only one type of event (one pitch) is present, that event is highly predictable or not surprising at all, and the entropy is 0. Entropy is maximised when the probability distribution over the events is uniform.

We are going to calculate entropy of 'features' extracted from the notes in monophonic lines. We will use features related to pitch and duration of the notes. A lot of features are possible: MIDI pitch number, MIDI interval, pitch contour, pitch class, note duration, inter onset interval etc. (see [3] for more features). We will focus on the following two features in the model presented here:

1. Pitch Class (C): the 'name' of a note, i.e., the pitch irrespective of the octave, such as C, D, etc. Represented as MIDI pitch modulo 12.
2. Inter Onset Interval (O): the time interval between the onset of consecutive notes in a sequence. Classes are derived by discretisation (an IOI is given its own class if it is not within 10% of an existing class).

We can measure the entropy of a given feature (H_C, H_O) in a sequence of notes by extracting note events, and calculate entropy from the frequencies of these events. As the features are meant to be calculated from successive single musical events (rather than arbitrary polyphonic structures), we will apply these measures only to skylined tracks or channels.



	H_C	H_O	$H_{C,O}$
Top voice	2.128	1.842	2.807
Lower voice	1.906	0.0	1.906

Fig. 1. Measuring entropy of the two voices in the first two measures of the first movement of Mozarts piano sonata KV 545

Entropy is also defined for a pair of random variables with joint distribution, allowing us to consider the joint events of a certain pitch class and a certain IOI class. We will denote the joint entropy measure of pitch class and IOI with $H_{C,O}$.

As an example of applying the complexity measures to music, consider Fig. 1. The three complexity measures just described have been applied in turn to the two musical lines found in the first two measures of a Mozart piano sonata. All three complexity measures show that the top voice contains the most information. In this case, most listeners would also agree that the upper voice is indeed the most interesting one, so this example supports our hypothesis that the most complex voice is experienced as foreground.

As a second example, consider four bars of pop music. Typically a solo instrument (voice, saxophone) will be playing a melody or a theme on top of a number of accompaniment instruments (rhythm guitar, drums). While the rhythm section will typically play short repeated patterns (e.g. 1-2 bars), the melody line often performs longer passages before it repeats itself (e.g. longer than four bars). When measuring information with entropy within the four bars, the voice playing repeated patterns doesn't add any new information for each repeat – the entropy measured on all repetitions is equal to the entropy of the pattern itself as the distribution of events is the same. In this sense the entropy measure is capable of taking habituation into account. As accompanying figures are often simple repeated patterns, the instrument playing longer phrases is likely to be considered more complex.

4 Reducing a MIDI File

The reduction algorithm is straightforward. First each voice (recognised by MIDI track and MIDI channel) is skylined in order to get monophonic sequences. All notes that do not make it through this process are marked. Then a window is slid over the notes in small steps. From the notes not marked by the skyline algorithm in each voice segment in each window, an interestingness value is calculated and stored. After this first pass, a threshold for including or excluding notes is

introduced. We only want to keep the notes occurring in the most interesting e.g. 50 % of the segments in the windows. By sorting all interestingness values, we obtain the actual threshold value by picking the value positioned at the wanted delimiting point (e.g. the median value).

Again, each voice segment in each window is considered. If the interestingness value of the note sequence is above the threshold, these notes are marked with the 'write' label. In the end, all notes that occur in any voice segment having an interestingness value higher than the limit will be given the write label.

The reduced MIDI file can now be written. Only notes marked with the 'write' label are written. For each file, we report the percentage of notes that – after skylining – were deleted by the reduction mechanism. As the search engine does not index notes that disappear by skylining, this is the actual reduction gain.

The window is determined by a fixed time interval, as complexity arises and is experienced over time. A note sequence stretched over a long time (say, one note per 5 seconds) could be complex, but will not be experienced as such, as only a simple fraction of it will be in our perceptual awareness at a time.

In section 5 we report how this basic algorithm fares when evaluated in a Query-by-Humming (QBH) context. We also examined one variant of the algorithm. Instead of using just one measure of interestingness, we can in turn mark notes using different interestingness measures – e.g. mark notes with a rhythmic (duration) measure as well as a pitch based measure, and write notes that have been marked by any of the measures. This approach will then mark notes that are interesting either because of rhythm or pitch.

5 Experiments

For a music search engine, the reduction of MIDI files can be an advantage, if the database size can be reduced noticeably while not losing any true positives for typical queries. At least the number of lost true positives should be as small as possible – any reduction of the number of true positives should be smaller than the relative reduction of the database size.

For the 1000 MIDI files in the MIREX dataset, files matching five queries have been identified as a ground truth prior to the experiments, and we will compare the lists of files retrieved when querying the untouched and the reduced datasets with these particular queries. If the search engine can retrieve the same, and in particular the correct files, for these queries, it would be a sign that the reduction actually removed uninteresting material.

5.1 The Data

The ground truth was established as part of MIREX 2006 as follows: All ten participating algorithms were used to retrieve the ten most similar items for every query. This resulted in a pool of up to 100 retrieved items per query. Human graders then manually compared the queries to the retrieval results and assigned similarity scores to each query-retrieval result pair. Both a rough score

Table 1. The ground truth for the MIREX 2006 "Karaoke" collection created by running 10 algorithms, having human graders assign similarity scores, and averaging and sorting the retrieved items by rough score. Items with equal average rough scores are grouped together, here shown with parentheses.

Query	Ground truth
q000001	(k001051,k000061,k000258)
q000002	(k000589),(k004517,k092387,k096238)
q000003	(k000589),(k004517,k093757)
q000004	(k095188,k095520)
q000005	(k094777),(k096523)

(1 = very similar; 2 = somewhat similar; 3 = not similar) and a fine score (a number in the range from 1 to 10) were collected from the graders. An overall ground truth was then established by calculating the average rough score for every retrieved item, throwing away items with an average score worse than 2 ("somewhat similar"), and finally grouping those items together whose average rough score was the same. This resulted in an ordered ranked list, where some items have equal ranks (see Table 1).

Query q000002 and q000003 refer to the same song, but q000003 is a more challenging, or 'badly sung' query. This explains the large overlap in the ground truth: the human judges agree that for both queries, the MIDI file k000589 contains a melody which is very similar to the query. However, only the items which are found in the ground truth for both queries, k000589 and k004517, really contain the query melody (ABBA's 'The winner takes it all'), and these files are almost identical arrangements of the song. Items k092387, k092387, and k096238 refer to other and quite different songs. We do not see the similarity the human judges claimed to see, and therefore we believe that one should take the ground truth for Queries q000002 and q000003 with a grain of salt. The ground truth for the other queries is more credible since for each of the remaining queries, all items in the ground truth are cover versions of the same piece.

5.2 Results

Four different reductions of the dataset were carried out. Building the index of the 1000 reduced files took for each reduction 2-3 days on the fastest machine at our disposal, so we decided to do only a limited number of experiments.

Table 2. Details of the four reductions

Reduction	Measure	Window size	Hop size	Threshold	Notes removed
1	$H_{C,O}$	4 sec	1 sec	0.5	27.15 %
2	$\frac{2}{3}H_{C,O} + \frac{1}{3}H_O$	4 sec	1 sec	0.6	36.71 %
3	H_C, H_O, $H_{C,O}$	4 sec	1 sec	0.5	15.50 %
4	$\frac{2}{3}H_{C,O} + \frac{1}{3}H_O$	10 sec	0.8 sec	0.5	35.34 %

Table 3. Retrieval results on the original data and on four different reductions. True positives are printed in *italics.*

	Reduction				
Query	None	1	2	3	4
q000001	*k000258,* *k000061,* *k001051*	*k000258,* *k000061,* *k001051*	*k000061,* *k000258,* *k001051*	*k000258,* *k000061,* *k001051*	*k000258,* *k000061,* *k001051*
q000002	*k004517,* *k000589,* k000076, k002413, k000034, k092493, *k092387,* k092386, k000843, *k096238*	*k004517,* *k000589,* k000836, k092948, k000076, k000034, k000861, k092493, *k092387,* k092386, k000843, *k096238*	k000836, k092948, k000076, k000916, k001012, k000034, k000861, *k092387,* k092386, k000843, *k096238,* ...	*k004517,* *k000589,* k000836, k092948, k000076, k000034, k000861, k092493, *k092387,* k092386, k000843, *k096238*	*k004517,* *k000589,* k000836, k092948, k000076, k000034, k000861, k092493, *k092387,* k092386, k000843, *k096238*
q000003	*k004517,* *k000589,* k000780, ...	k000780, k000899, k096248, k000116, k000050, k092960, k092959, k000699, k000881, k000270, k092948, k000092, k000392, k000234, k092442, ...	k000780, k000899, k096248, k000116, k000050, k092960, k092959, k000699, k000881, k000270, k092948, k000092, k000392, *k000589,* *k004517,* ...	k000780, k096248, k000116, k000050, k092960, k092959, k000699, k000881, k000270, k092948, k000092, k000392, *k000589,* *k004517,* ...	k000780, k000899, k096248, k000116, k000050, k092960, k092959, k000699, k000881, k000270, k092948, k000092, k000392, k000234, k092442, ...
q000004	*k095188,* *k095520*	*k095188,* *k095520*	*k095520,* k000811, ...	*k095188,* *k095520*	*k095188,* *k095520*
q000005	*k094777,* *k096523*	*k094777,* *k096523*	*k094777,* *k096523*	*k094777,* *k096523*	*k094777,* *k096523*

One reduction was done using the joint pitch class and IOI entropy measure $H_{C,O}$. Two reductions used a linear combination of $H_{C,O}$ and H_O – giving a little more weight to rhythmic differences. In one experiment (reduction 3), the three measures were used in turn to mark the interesting notes as explained in chapter 4. The threshold was set to 50 % or 60 % (discarding interestingness values under this limit). Table 2 displays information about which measures,

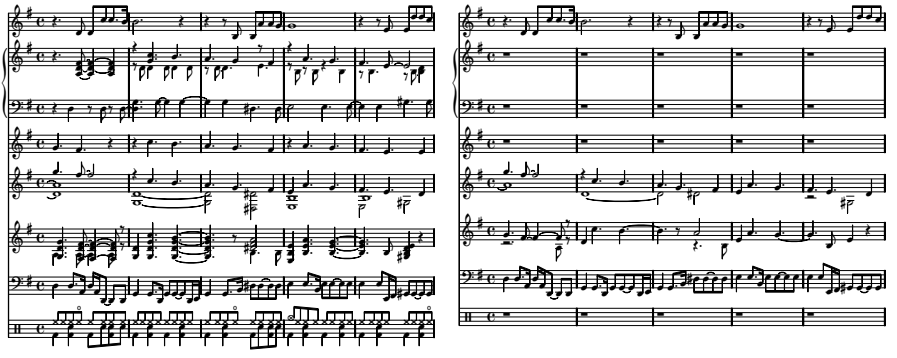


Fig. 2. Five measures of an ABBA song, the original on the left and the summary on the right

window sizes, and thresholds were used, as well as the resulting reduction rate (averaged over all files). We notice that the different parameters result in different reduction rates (the percentage of notes removed by the interestingness measure after skylining).

The indexes were then queried with the five queries. Table 3 displays the ranked lists returned by the retrieval algorithm with the original data and the four different reductions of the data.

The queries q000001, q000004, and q000005 retrieved the same files in all datasets (except in the highly reduced reduction 2). Queries q000002, and q000003 (that refer to the same song) resulted in some changes in the ranking delivered by the search engine, and some of the ground truth songs were ranked very low.

The fourth reduction used the same complexity measure as reduction 2, but a longer (10 sec) window (and a threshold of 0.5). This eliminated 35.34 % of the notes on average over all files in the Karaoke dataset. For four of the queries, the same songs could be retrieved from the unprocessed and the summarised dataset. For the remaining query (the challenging q000003), the ground truth files were not retrieved from the reduced dataset, but when listening to the reduced files, they seem to contain all instances of the melody that the query refer to, so the hit in the unreduced song might have been due to a lucky match to some unintended part of the song. After all the query is a poor interpretation of the ABBA song.

Also for this reduction, nine of the ten songs retrieved for query q000002 on the unreduced data appear among the 12 first matches for the same query in this fourth reduction. This tells us that indeed many of the important and interesting passages in the music files are preserved.

When listening to reduced files, it becomes apparent that the original arrangements are highly recognisable. And that despite the files have been both skylined and reduced by the interestingness approach. In fact, after skylining and the interestingness reduction, 58.21 % of the notes on average per file in

the fourth reduction have been removed. Most of the important material is still there - mainly accompaniment figures and drum loops are deleted.

Fig. 2 shows on the left five bars from one of the ground truth files (k000589) in the Karaoke dataset (transcribed by the authors). On the right the same passage is depicted after applying the skylining and reduction using the settings from our fourth experiment. The top staff containing the usually sung melody is kept intact. The second voice (next two staffs) holds a piano accompaniment that was completely omitted, probably due to the fact that it was rhythmically uninteresting as there is an onset on every eighth note. The highly active bass line (second last staff) was kept, and the very repetitive drum track was removed.

6 Conclusion and Future Work

We presented an approach to identifying interesting melodic lines in MIDI files. A MIDI file reduction algorithm based on this approach was presented and evaluated on a QBH task. The current results with a limited evaluation scenario suggest that our approach is a promising path to follow. Future work include testing the approach on more data.

The reduction algorithm was explicitly designed for the QBH task, optimised to preserve interesting melodic lines in MIDI files. Although we (immodestly) find that the summarised files sound like good representations of the originals, it would be interesting to explicitly seek to conserve the *structure* of the composition e.g. in terms of harmonic content (for example, in the current approach it is not guaranteed that the bass line will be kept). This could be useful when summarising MIDI files to be played on hardware that can only play a limited number of notes simultaneously – e.g. to automatically make a standard MIDI file comply with the Scalable Polyphony MIDI file format [7]. This is a different reduction goal where we also think the measures of interestingness can play an important role.

Acknowledgments. This research was supported by the Austrian Research Fund (FWF), project number M1027-N15 (Combining Audio and Symbolic Approaches for Music Similarity Search) and project P19349-N15 (Computational Performance Style Analysis from Audio Recordings), and by the Austrian Federal Ministries of Education, Science and Culture and of Transport, Innovation and Technology.

References

1. Chai, W.: Melody retrieval on the web. Master's thesis, MIT (2001)
2. Cilibrasi, R., Vitányi, P., de Wolf, R.: Algorithmic clustering of music based on string compression. Computer Music Journal 28(4), 49–67 (2004)
3. Conklin, D.: Melodic analysis with segment classes. Machine Learning 65(2-3), 349–360 (2006)

4. de Nooijer, J.: Cognition-based Segmentation for Music Information Retrieval Systems. Master's thesis, Utrecht University (2007)
5. Dubnov, S., McAdams, S., Reynolds, R.: Structural and affective aspects of music from statistical audio signal analysis. Journal of the American Society for Information Science and Technology 57(11), 1526–1536 (2006)
6. Li, M., Sleep, R.: Genre classification via an LZ78-based string kernel. In: Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR 2005), London, U.K. (2005)
7. Lui, S., Horner, A., Ayers, L.: MIDI to SP-MIDI transcoding using phrase stealing. IEEE Multimedia 13(2), 52–59 (2006)
8. Madsen, S.T., Widmer, G.: A complexity-based approach to melody track identification in midi files. In: Proceedings of the International Workshop on Artificial Intelligence and Music (MUSIC-AI 2007) held at the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), Hyderabad, India (January 2007)
9. Madsen, S.T., Widmer, G.: Towards a computational model of melody identification in polyphonic music. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), Hyderabad, India (January 2007)
10. Meyer, L.B.: Emotion and Meaning in Music. The University of Chicago Press, Chicago (1956)
11. Orr, M.G., Ohlsson, S.: Relationship Between Complexity and Liking as a Function of Expertise. Music Perception 22(4), 583–611 (2005)
12. Ruppin, A., Yeshurun, H.: MIDI Music Genre Classification by Invariant Features. In: Proceedings of the 7th International Conference on Music Information Retrieval, Victoria, BC, Canada, October 2006, pp. 397–399 (2006)
13. Snyder, B.: Music and Memory: An Introduction. MIT Press, Cambridge (2000)
14. Typke, R.: Music Retrieval based on Melodic Similarity. PhD thesis, Department of Information and Computing Sciences, Universiteit Utrecht, Netherlands (2007)
15. Uitdenbogerd, A.L., Zobel, J.: Manipulation of music for melody matching. In: ACM Multimedia, pp. 235–240 (1998)

Automatic Synchronization between Audio and Partial Music Score Representation

Antonello D'Aguanno and Giancarlo Vercellesi

Laboratorio di Informatica Musicale (LIM)
Dipartimento di Informatica e Comunicazione (DICo)
Universitá degli Studi di Milano,
Via Comelico 39,
I-20135 Milano, Italy
{daguanno,vercellesi}@dico.unimi.it
http://www.lim.dico.unimi.it

Abstract. In this paper we propose an algorithm namely ParSi (Partial Synchronization) which aligns different PCM executions with a partial music score representation. Generally, most synchronization algorithms require a complete score representation; despite of them, our algorithm is able to synchronize different PCM audio executions with a partial music score codified in MIDI. Therefore, only one MIDI instrument -chosen by user- is used during audio-score alignment. The audio analysis is performed using a notch filter. A set of conveniently parameters are used during the decisional phase. The analysis of synchronization results is provided. It shows we can have a good automatic synchronization even if just a partial score is available.

Keywords: MIR, MIDI, synchronization, alignment, score following.

1 Introduction

Contemporary digital music archives consist of huge collections of heterogeneous documents. For a music piece, an archive may contain corresponding scores in different versions, for example, voice and piano or orchestral, as well as several interpretations, e.g. played by different performers and recorded in diverse formats (CD recordings, MP3, FLAC and so on). The heterogeneity of music information makes retrieval hard to accomplish [1] [2] [3] and as a consequence many problems remain unsolved [4]. One important problem that needs a solution is synchronization, which requires the implementation of algorithms that automatically link different audio streams of the same piece to symbolic data formats representing the different scores.

The automatic score-audio alignment can be useful for musicologists who want to investigate agogics and tempo [5]. In addition, temporal linking of score and audio data can be useful for automatic tracking of score positions during a performance. Such algorithms may be also useful for score-following applications [6]. These possibilities represent a useful tool for music students, who can listen to music audio and, at the same time, see the corresponding notes.

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 100–111, 2010.

The state of the art proposes algorithm which generally requires a complete score representation. Unfortunately, we may have many cases where a partial score is available. For this reason in this paper we describe an algorithm which would like to propose a possible solution to that situations. The ParSi (Partial Synchronization) algorithm is able to synchronize different PCM audio executions using a partial music score at symbolic level. The score has to be codified in MIDI. therefore, only one MIDI instrument -chosen by user- is used during audio-score alignment.

This article is organized as follows: section 2 presents an overview of score following and synchronization algorithms proposed in literature. Section 3 describes the ParSi algorithm. The analysis of results is discussed in section 4. Finally, conclusions will be summarized in 5.

2 Related Works

Many algorithms have been proposed in literature that deal with synchronization. The majority of them can be subdivided into two groups: in the first one, audio and score have to be analysed, while the second one requires the realisation of correct links between these two layers ([7] [8] [9] [10] [11] [12] [13] [14]). The algorithms proposed in the literature use several different systems to implement audio analysis, with well-known tools from audio signal processing. For example, [12] uses a Short Time Fourier Transform, in [8] proposes an onset detection followed by pitch detection. In [11] the feature extraction procedure performs these operations: decomposition of the audio signal into spectral bands corresponding to the fundamental pitches and harmonics, followed by the computation of the positions of significant energy increases for each band - such positions are candidates for note onsets. Other popular solutions proposed in the literature to select the correct links between audio and score are based on the *template matching technique* ([12] [7] [13]). Such algorithms build a MIDI score to obtain a template of the real execution, which is compared to the real audio using a DTW[1] programming technique [15]. The correct synchronization is then obtained from the difference between the agogics of the real execution and that of MIDI.

The algorithm described here uses an approach based on recursive research with notch filter for audio analysis and a partial score event extraction from MIDI coding.

3 The ParSI Algorithm

The algorithm proposed here is able to synchronize a MIDI score with one or more PCM audio executions using only one instrument codified in the MIDI file; this instrument has to be selected by the user. Therefore, we do not try to

[1] Dynamic Time Warping is a technique for aligning time series that has been well known in the speech recognition community since the 1970s.

synchronize the whole score with the audio but we extract score informations from one instrument in order to automatically align it with the audio execution.

ParSi algorithm consists of three different phases (figure 1):

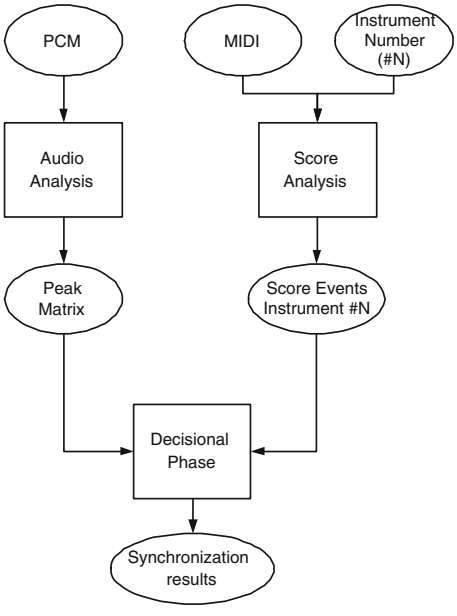


Fig. 1. The ParSi architecture

- in the first phase, the selected instrument codified in the MIDI score is read in order to extract all relevant score events.
- in the second phase, the PCM audio signal is analysed driven by the extracted score events, in order to identify all possible time-attacks notes in terms of energy peaks.
- in the third and last phase, a decisional matching is performed to relate the event at the score level of the selected instrument with the same event at the audio level. The decisional method is based on the attack time of notes extracted at audio level.

Each phase is described in the following sections.

3.1 The MIDI Score Analysis

In this phase, the selected instrument codified in the MIDI file is read to extract the score events as illustrated in figure 2.

For each measure, only notes with strong accent are selected because it is easier to recognize them at audio level. Thus the score is represented in a suitable manner for synchronization. For each strong accent each note is codified with

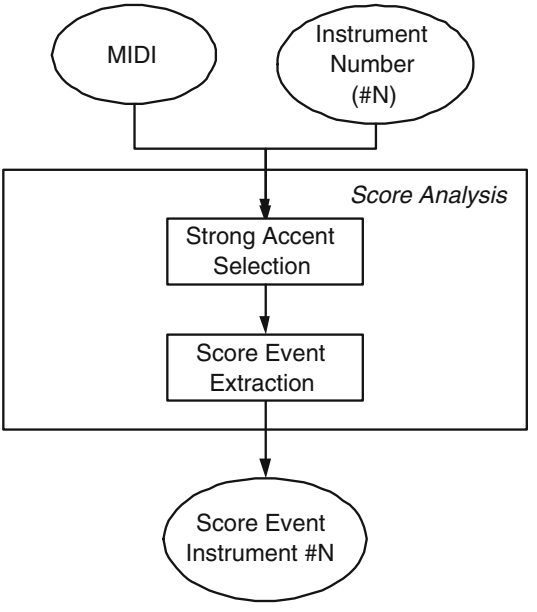


Fig. 2. Score Analysis Flowchart

its fundamental frequency, the duration in MIDI ticks, the beat and measure number. Notes grouped between two asterisks (∗) correspond to a chord. A vertical rest is represented with −1. An example is shown in the table below (1); the corresponding score is shown in figure 3:

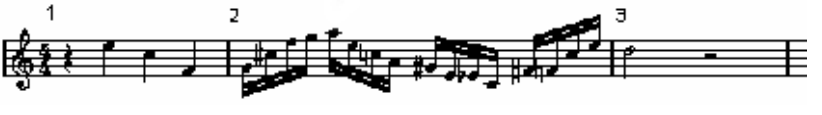


Fig. 3. The score represented in table 1

3.2 The PCM Audio Analysis

In the audio analysis, a recursive, second-order notch filter bank [16] is used as shown in figure 4.

The aim of this phase is the detection of each attack-time for each note codified in the selected instrument at score level. For each verticalized score event (see table (1)), the audio signal is filtered with a notch filter centered both on the fundamental frequency and on the first four harmonics of the examined note. The portion of signal analyzed is calculated defining a conveniently cluster position and size.

The cluster position is obtained as follow. Let T_{track} and T_{score} be the audio track duration in seconds and the MIDI track duration in MIDI ticks, respectively. We calculate the tick duration $T_{tick} = T_{track}/T_{score}$. The cluster position

Table 1. Example of a single instrument coding

Fundamental Frequencies	Durations	Beat	Measure
*			
-1	480	1	1
*			
659	480	1	2
*			
523	480	1	3
*			
349	480	1	4
*			
392	480	2	1
*			
880	480	2	2
*			
415	480	2	3
*			
370	480	2	4
*			
698	960	3	1
*			
-1	960	3	3
*			

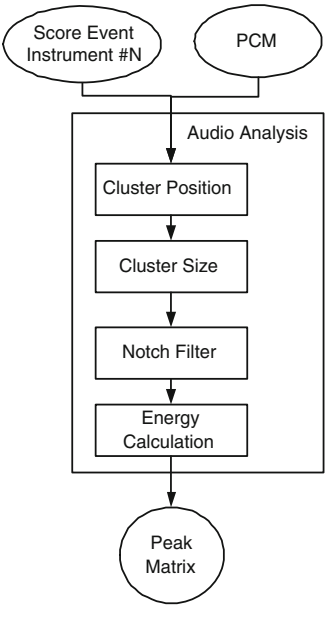


Fig. 4. Audio Analysis Flowchart

is determinated as $T_{cluster} = N_{tick}/T_{tick}$ where N_{tick} is the sum of ticks associated to the previous notes.

The cluster size is calculated as $Cluster\_size = ((T_{track}/N_{events}) * variation$ where N_{events} is the total event number extracted at score level; $Variation$ is an input parameter -related to the interpretation- which represents the amount of tempo variation of audio execution with respect to the MIDI performance. $Variation$ lies on a range from 1 to 4 where 2 means we have not meaningful variation. In other words, this parameter allows to increase or decrease the $cluster\_size$ with respect to the amount of expected tempo changing.

After notch filter, we calculate the energy envelope in order to generate the so called *peak matrix*. After energy computation we calculate its peaks with respect to the score notes. Then, all the possible retrieved peaks are stored into peak matrix as shown in table 2. The first column contains the analyzed fundamental frequency; the matrix row number is equal to the 'number of events extracted at the score level' N_{events}. The second column contains a list of all possible peak times (in seconds). If no peak is retrieved, the matrix is filled with 0.

Table 2. An example of *peak matrix*

Fundamental Frequencies	Retrieved Peak Time
659	$T_1^1, T_2^1, ...T_n^1$
523	$T_1^2, T_2^2, ...T_n^2$
349	$T_1^3, T_2^3, ...T_n^3$
....	....

An example of Notch filtering applied on fundamental frequency is shown in figure 5. Consider note A at 440Hz. While (a) shows its waveform signal, this signal is filtered with the notch filter with center frequency 440Hz (b), which yields a filtered signal with high energy. Note that if the signal is filtered at different center frequencies, the signal will contain less energy: see the cases for center frequency at 220Hz (c) and 880Hz (d).

To be able to retrieve every note, avoiding superimposition between adjacent semitones, the bandwidth of the notch filter is dependent to the analyzed frequency with respect to this formula: $bandwidth = 3 * 2^{\log_{10}(f)}$ where f is the frequency in Hertz.

3.3 The Decisional Phase

The last step of ParSi algorithm is the decisional phase where the time-alignment between audio and score events is created (figure 6). As we said before, only score events belonged to only one instrument (conveniently chosen by user) is considered during automatic synchronization.

We start invoking the **init** function which initializes some variables used during the algorithm: SS_{size}, $N_{virtual\_peaks}$ and $tolerance$:

- the maximum size of subsequence SS_{size} is provided as input by user. This value is used to define the subsequence limits where our algorithm will extract the candidate synchronization sequences;

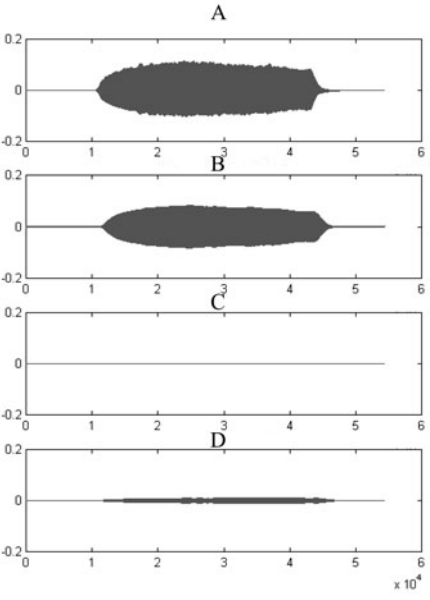


Fig. 5. Spectrum examples of a 440Hz signal (a) filtered with a notch filter having its center frequency at 440Hz (b), 220Hz (c) and 880Hz (d)

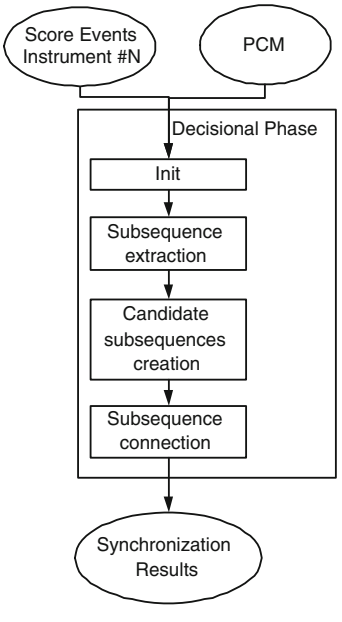


Fig. 6. Decisional Phase Flowchart

- $N_{virtual\_peaks}$ is the number of virtual peaks. They are inserted into the matrix peak when a 0 is found. To avoid bad alignment results, this number must not be over the 35% of subsequence length SS_{size};
- the *tolerance* range is used to evaluate the reliability of retrieved event time at audio level with respect to score event time in MIDI ticks. The default range lies between 0.4 and 0.8.

Then we define a subsequence region by means SS_{size} value and we invoke the **subsequence extraction** function. Aim of this function is to find the more conveniently values for *tolerance* and $N_{virtual\_peaks}$ in order to retrieve all possible valid synchronization events which will be inserted into the candidate subsequence. We may have the following situations:

- we do not find an acceptable subsequence. In this case we increase the tolerance range of one third: $\Delta\_tolerance = \Delta\_tolerance + 1/3$. If this operation is not sufficient, we increase the virtual peak number.
- we find too much acceptable subsequences. In this case we decrease the tolerance range of one third: $\Delta\_tolerance = \Delta\_tolerance - 1/3$.

In case we found a sequence of chords, it is decomposed in all the possible monody sequences.

The output of this function provides a two or three dimension matrix which contains, for each note/chord belonging to the considered subsequence, the possible synchronization events which respect the tolerance. If we get the limit of *tolerance* and $N_{virtual\_peaks}$, and no acceptable subsequences are found, we fill the matrix with 0.

At the end of this phase, the **candidate subsequences creation** function is invoked. Its objective is to select the most reliable synchronization events given in output by **subsequence extraction** function. The candidate subsequence is filled selecting, for each note, the peak which has the lowest distance between the peak time position in seconds retrieved at audio level and the time position in ticks calculated at score level.

Table 3. An example of subsequence connection

CS #1	CS #2	CS #3	CS #4	FS
0.00				0.00
0.50				0.50
1.15	0			1.15
1.91	0			1.91
	0			2.09
	0			2.66
	0	3.25		3.25
		3.54		3.54
		4.25	3.90	4.075
		4.60	4.32	4.46
			4.88	4.88

The final step is the **subsequence connection** where we create the final sequence. Peak for peak, we build the final sequence linking together the candidate subsequences as follow:

- if we have a single candidate event greater than 0, and its value is greater than the previous, this value is inserted into the final sequence;
- if we have two or more candidate events greater than 0, the arithmetic mean of these values is calculated;
- if we have all the candidate events equal to 0, this value is fitted by linear interpolation.

We end the decisional phase generating the synchronization results.

4 Experiments and Results

A prototype of our synchronization algorithm has been implemented in C++. In this section experiments and results are provided. Tests were run on an Intel Pentium IV, 2.6 GHz with 512 MByte RAM under Windows XP SP2.

To perform our objective tests, 31 pieces of music have been considered. They have a length from 30 seconds to 416 seconds.

They can be grouped in:

- polyphonic and polytimbric MIDI pieces;
- polyphonic piano pieces;
- opera pieces (with human voice);
- symphonic pieces (without human voice);
- pop pieces.

The complete list can be found at the website www.lim.dico.unimi.it/parsilist.html.

The results given by ParSi algorithm have been compared with manual synchronization of the same pieces. We provide the objective evaluation in term of percentage of *right measures* (figure 7).

We have a right measure if all synchronization events of the measure are right. A synchronization event retrieved by ParSi algorithm ($se_i^{automatic}$) is right if it does not come before or does not come after the previous and the next synchronization events obtained by manual synchronization (respectively se_{i-1}^{manual} and se_{i+1}^{manual}).

$$se_{i-1}^{manual} < se_i^{automatic} < se_{i+1}^{manual}$$

We can consider this criteria enough for applications where an accurate and precise synchronization is not required.

In order to obtain the best synchronization results, the selected instrument has been selected in order to get available the most score events number. Furthermore, when possible, we have selected the piano instrument because the state of the art shows (see section 2) it is generally easier to synchronize.

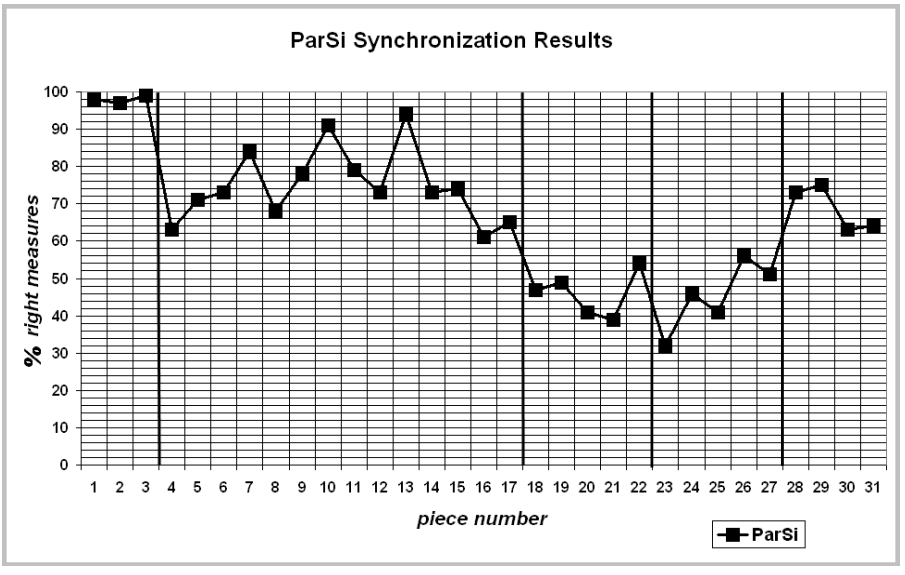


Fig. 7. ParSi Synchronization Results

Experiments show that ParSi algorithm synchronizes correctly (96% - 100%) polyphonic and polytimbric MIDI pieces (piece numbers from 1 to 3).

ParSi synchronization of polyphonic piano pieces (numbered from 4 to 17) gives interesting results, with a percentage of success between 60% and 95%.

Opera pieces (numbered from 18 to 22) have a worse synchronization precision (35% - 60%) than piano pieces because of the more complexity of audio signal. We have a similar situation with symphonic pieces (pieces numbered from 23 to 27). Here, we have the worst case for the piece number 23 ('Arabic Dance' from *The Nutcracker* by Tchaikovsky) because of several repeated notes which generate similar time peaks.

Finally, pop music (pieces numbered from 28 to 31) provides a interesting percentage of success which lies between the and 55% and 70%.

If we compare these results with other works which requires a complete score ([8] [9] [10] [11] [12] [14]), we can state ParSi provides good automatic synchronization results with a partial score representation.

5 Conclusions

In this work we have proposed an algorithm to automatic synchronize different PCM audio performance to a partial music score codified in MIDI. We use a notch filter during audio analysis and a set of conveniently parameters to perform the decisional phase. Of course, the instrument selection is the most critical decision and it could deeply affect the quality of synchronization results. An instrument with the most number of events is recommended.

Analyzing the experimental results we can state that the ParSi approach provides interesting results, even if a partial score is used. The pop music test is the more interesting experiment because, generally, incomplete scores are available. The synchronization results of opera and symphonic pieces should be improved; in fact future works are mainly addressed to algorithm improvement for a better time-alignment of that music genres.

References

1. Stephen Downie, J.: Music information retrieval. Annual Review of Information Science and technology, ch. 7, vol. 37, pp. 295–340. Blaise Cronin, Medford (2003)
2. Wiering, F., Veltkamp, R.C., Typke, R.: A survey of music information retrieval systems. In: 6th International Conference on Music Information Retrieval, ISMIR 2005, London, UK, September 11-15, pp. 153–160 (2005)
3. Foote, J.: An overview of audio information retrieval. Multimedia Systems 7(1), 2–10 (1999)
4. Clausen, M., Kurth, F., Müller, M., Ribbrock, A.: Content-based retrieval in digital music libraries. In: Heery, R., Lyon, L. (eds.) ECDL 2004. LNCS, vol. 3232, pp. 292–303. Springer, Heidelberg (2004)
5. Bainbridge, D., Dewsnip, M., Witten, I.H.: Searching digital music libraries. In: Lim, E.-p., Foo, S.S.-B., Khoo, C., Chen, H., Fox, E., Urs, S.R., Costantino, T. (eds.) ICADL 2002. LNCS, vol. 2555, pp. 129–140. Springer, Heidelberg (2002)
6. Schwarz, S., Orio, D., Lemouton, N.: Score following: State of the art and new developments. In: Proceedings of the Conference on New Interfaces for Musical Expression, pp. 36–41 (2003)
7. Dannenberg, R.B., Hu, N.: Polyphonic audio matching for score following and intelligent audio editors. In: Proceedings of the 2003 International Computer Music Conference (2003)
8. Arifi, V., Clausen, M., Kurth, F., Muller, M.: Automatic synchronization of music data in score-, midi- and pcm- format. In: 4th International Conference on Music Information Retrieval, ISMIR 2003 (2003)
9. Hewlett, W., Selfridge-Fields, E. (eds.): Automatic Synchronization of Musical Data: A Mathematical Approach. Computing in Musicology. MIT Press, Cambridge (2004)
10. Dixon, S., Widmer, G.: Match: A music alignement tool chest. In: 6th International Conference on Music Information Retrieval, ISMIR 2005 (2005)
11. Muller, M., Kurth, F., Roder, T.: Towards an efficient algorithm for automatic score-to-audio synchronization. In: 5th International Conference on Music Information Retrieval, ISMIR 2004 (2004)
12. Soulez, F., Rodet, X., Schwarz, D.: Improving polyphonic and poly-instrumental music to score alignment. In: 4th International Conference on Music Information Retrieval, ISMIR 2003, pp. 143–148 (2003)
13. Turetsky, R.J., Ellis, D.: Ground-truth transcriptions of real music from force-aligned midi syntheses. In: 4th International Conference on Music Information Retrieval, ISMIR 2003 (2003)

14. D'Aguanno, A., Vercellesi, G.: Automatic synchronisation between audio and score musical description layers. In: Falcidieno, B., Spagnuolo, M., Avrithis, Y., Kompatsiaris, I., Buitelaar, P. (eds.) SAMT 2007. LNCS, vol. 4816, pp. 200–210. Springer, Heidelberg (2007)
15. Rabiner, L.R., Juang, B.H.: Fundamentals of Speech Recognition. Prentice-Hall, Englewood Cliffs (1993)
16. Oppenheim, A.V., Schafer, R.W.: Discrete-time signal processing. Prentice-Hall, Englewood Cliffs (1989)

Automatic Image Tagging Using Community-Driven Online Image Databases

Marius Renn, Joost van Beusekom,
Daniel Keysers, and Thomas M. Breuel

IUPR Group, Technical University of Kaiserslautern, Germany
m_renn@informatik.uni-kl.de, joost@iupr.dfki.de,
keysers@iupr.com, tmb@informatik.uni-kl.de
http://www.iupr.org

Abstract. Automatic image tagging is becoming increasingly important to organize large amounts of image data. To identify concepts in images, these tagging systems rely on large sets of annotated image training sets. In this work we analyze image sets taken from online community-driven image databases, such as Flickr, for use in concept identification. Real-world performance is measured using our flexible tagging system, *Tagr*.

1 Introduction

With the rise of the internet and the rapid growth of storage, a number of large image databases have emerged on the web. Fueled by the popularity of low-cost digital image capturing devices and the web 2.0 trend of dynamic community driven websites, many of these databases consist of photos submitted by community members. As our observations using the Flickr API[1] show, these websites may see growth rates of over one million photo submissions per day (see Figure 1). Likewise, offline personal photo collections now often contain thousands of photos, that can be stored and viewed on high resolution computer screens at nearly zero cost.

Due to this rapid growth of image content both on- and offline, it has become increasingly difficult to organize these massive amounts of visual data. To overcome this difficulty, many photo sharing websites and modern offline photo organizing software applications allow the user to add textual annotations to the images. These annotations usually consist of a list of keywords or *tags*, that describe some aspect of the image *content*, and allow organizing, searching and filtering images, using algorithms based on these keywords. Unfortunately, a great deal of images both on- and offline exist, that have no textual representation whatsoever. Asking humans to manually label such images is not only costly and time-consuming, but also poses privacy and security issues. Furthermore, the almost exponential growth of photos on community websites would require an ever growing team of labelers. Thus it is desirable to add missing tags to images automatically.

[1] http://www.flickr.com/services/api/

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 112–126, 2010.

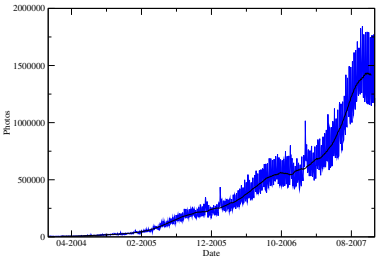


Fig. 1. The number of photos uploaded to Flickr measured on a daily basis from January 2004 until November 2007, using the Flickr API

Although computers are still a long way from identifying and textually describing image concepts in the way humans do, it is possible to train computers on large previously annotated image databases, in order to learn the associations between visual image data and their textual descriptions. Now that large online image databases are available free of charge, these provide an interesting alternative to professionally labeled commercial sets. Furthermore, many of these community-driven image collections are annotated with tags, submitted by the users.

In this work we make use of large online image databases, such as Flickr[2], and compare their performance to a professionally annotated image set. Our analysis will show where potential problems lie, and what can be done to overcome them. To evaluate performance we use our flexible automatic image tagging system, called *Tagr*.

In Section 2 we will give an overview of related literature. Section 3 describes the tagging system on which the measurements are based. The image sets are described in Section 4. Section 5 discusses the results. The paper is concluded by Section 6.

2 Related Work

In this work, our focus lies on the compared performance between commercial image sets and online community-driven databases as models for image tagging. To our knowledge, there has been no prior work on such a performance analysis for automatic image tagging. However, the development of image tagging itself has become a subject of interest in recent years, and numerous approaches to accomplish this task have emerged. We shall give a brief outline here. One popular method of automatic annotation is the association of keywords with image regions. These image regions can be extracted using common image segmentation techniques, as in [7], where a grid-based segmentation method to identify image regions is used, or using clustering methods as in [1], [2] and [5]. In [6] Li and Wang present a real-time automatic image annotation system called ALIPR,

[2] http://www.flickr.com

that uses advanced statistical modeling and optimization techniques to train computers various semantic concepts using example pictures. Their system is evaluated on manually selected images obtained from Flickr. Other systems, like the one used here, base annotations on the entire image scene, using global image features. In [12] nonparametric density estimation of global image features is used as a model for keyword probabilities. The Corel and Getty datasets are used for performance evaluation. As in our system, the authors in [11] describe an image tagging system, that uses content-based image retrieval at its core. Users interact with the system to produce correct high-level metadata. In this work we will use our own scene-based tagging system, called *Tagr*, to investigate the use of various annotated image sets as models for keyword probabilities. The high flexibility, speed, and access to the source code led to the choice of *Tagr* for these experiments.

3 System Overview

We now give a brief overview of *Tagr*, the tagging system used for our experiments. To meet our requirements of tagging a broad class of images while offering flexibility in the methods used, *Tagr* combines techniques from three, not strictly separate, domains of pattern recognition and machine learning:

- Hard-Coded Rules: *Tagr* uses simple rule-based image analysis to generate higher-level metadata, such as image dimensions, dominant colors, and image format.
- Machine Learning: To assign images to a certain category (such as `graphic` or `photo`), *Tagr* uses well-known classifiers on extracted image features.
- Content-Based Image Retrieval: In content-based image retrieval (CBIR), images in a database are searched by their content, and not by their textual representation (i.e. filename). *Tagr* uses CBIR to find the k closest matches of some query image in an image database (k-nearest-neighbor search), and requires some definition of a distance metric between the images or image features. *Tagr* requires the images in the database to have been previously supplied with tags, that each describe the image content. This way, *Tagr* is able to extract a set of tags for a given query image, by analyzing the most frequent tags among the nearest neighbors of the database.

At the heart of the classification and tagging process lies a *query tree*, in which an image query is handed to the root node and passed down to the children for further processing. Nodes may apply filters to the image, or process it in any other way. The leaf nodes are responsible for creating an initial textual annotation of the image. Most often we make use of the `FireNode`, which sends the image query to the *Flexible Image Retrieval* (short: *FIRE*) system for nearest neighbor comparison. *FIRE* is an image retrieval system originally developed by Thomas Deselaers of the RWTH Aachen, and now maintained by him and Daniel Kaisers. Interested readers can find more information about it in [4]. For a given query image, *FIRE* returns the k nearest neighbors of this image over a database

of model images, given a set of features and distance metrics for comparison. These images are returned as a scored list. In the query tree, the FIRE node simply sends the query image to a specified running FIRE server, and passes the result list back up the tree. This image list is then converted to a list of tags by the `ImagesToTagsNode`, which maps each image file to its textual description. The resulting tags obtain the scores of the images they resulted from. Usually this list is passed into a `PackResultsNode`, which combines equally named result strings by summing their scores.

3.1 Feature Selection

FIRE offers a variety of feature extraction methods and distance measures to be used in the nearest neighbor search. In the most basic configuration, we employ color histograms and Tamura texture [10] features, and the Jensen-Shannon divergence as a similarity measure. However, the flexibility of FIRE and the query tree approach allow us to test many other configurations of features and comparison methods. Although color and texture histograms produce convincing results, their downside is the loss of all spatial information. Therefore, a number of spatial features were tested, including various configurations of spatiograms [3]. The best results however were obtained using *weighted* histograms: These make use of the observation, that important classifying aspects of a photo usually lie in distinct regions on the image plane. For instance, the subject of a photo usually lies approximately in the center of the image. Color information close to the photo's boundaries, on the other hand, often shows other concepts, such as the ground or sky. To capture these distinct areas, we use a set of weighted histograms, one for each region in the image. In [9] a similar approach, called *fuzzy regions*, is used, where the image is subdivided into 5 regions. In our case, we use 3 weighted histograms to represent the top, center and bottom regions of the image.

4 Image and Tag Sets

Large pre-annotated image sets not only provide the model for the k-nearest neighbor search, but are also of use as test sets for evaluation. The given tag data of an image is compared to the tags returned by the system, which allows us to measure performance in terms of precision and recall. Table 1 gives an overview of the image sets used, along with their size (in number of images), how they were aggregated, and the most frequent tags.

4.1 COREL Set

To compare results of imagery from community-driven websites to those of commercial image sets, we randomly selected $26,803$ textually annotated photos from the Corel database, as a commercial representative. Each photo is tagged by 4 keywords on average, and there are a total of $4,900$ unique tags in the database. The main advantage of the Corel set is the fact, that the photos are professionally annotated, and thus exhibit consistency and objectivity.

Table 1. The annotated image sets used for our experiments

	COREL	Flickr	FotoCommunity	LabelMe
Images	26,803	52,478	20,834	32,025
Aggregation	commercially available	Flickr API	crawled website	download available
Top Tags	sky, water, people, trees, building	wedding, 2007, beach, nature, sky	motives, nature, people	car, head, tree, window, building

4.2 Flickr Set

Two prominent examples of online photo communities are Google's Picasa and Yahoo's Flickr, that provide millions of publicly available tagged photos. In this work, we chose to use Flickr's database, as an API for a variety of host languages is available. Using this API for Python, we implemented a number of tools, that allowed us to access the photo and tag data from the Flickr database. This was especially useful to accomplish the following tasks:

- Extraction: Given a number n, and a set of input tags T, download up to n photos and tags, that are annotated with the tags T', where $T \subseteq T'$.
- Tagging: As each image file from Flickr has a unique name, tag data can be added to previously downloaded photo sets.
- Sampling: Download random tags to sample keyword frequency.

Using these tools, first a set of typical tags was extracted from the Flickr database. These tags were analyzed to obtain a set of common topics, some of which are shown in Table 2. Using these topics as tag search words, a set of $52,478$ photos along with their full set of user annotations were downloaded from Flickr. The advantages of extracting photos from a large online database are the flexibility in quantity and scope. For instance, in order to compare results on the Corel set to a similar set, the Flickr API was utilized to download an image database with approximately the same size and same tags as the Corel set. This allowed us to test classifiers, that were trained on the Corel set, on a similarly labeled set of different photos.

Table 2. An excerpt from the categories (top row) used to download Flickr image sets. These categories were evaluated by looking at common tags and subjects depicted in the photos.

animal	event	food	nature	people	sports
cat	concert	fruit	beach	face	golf
dog	party	vegetable	forest	person	hockey
farm	wedding	fastfood	sky	portrait	tennis

4.3 FotoCommunity Set

A different approach to annotated image aggregation was taken, by crawling the German online community site `fotocommunity.de`. This website's intended contributors are amateur and professional photographers, who can get help and tips from other photography enthusiasts. Photos tend to be more professional and artistic than on Flickr. Although the photos are not tagged, they have been categorized (by the site's maintainers) well enough, that the category hierarchies themselves can be used as textual annotations. The category tree, downloaded from the website by our web-crawler, has over 950 nodes, with a total of 740 leaf nodes (most detailed categories). A set of 20,834 photos distributed among all categories was downloaded for the tagging model.

4.4 LabelMe Set

Finally, we gathered image collections from websites, that also focus on textually annotating image data. The *LabelMe* project[3] from the Massachusetts Institute of Technology aims to collect contributions from many people to build a large high quality database for research on object recognition. Instead of merely supplementing given images with keywords, users trace the boundaries of objects in images, and add labels to these regions. Each time an object is labeled, the data is continuously saved and made immediately available to interested researchers[4]. A total of 32,025 labeled images were downloaded. The most common object tags with frequencies of at least 100 occurrences were extracted and assigned to the images they occurred in. The segmentation information itself was ignored. This resulted in a list of 116 distinct tags. Images that did not contain objects with these tags were filtered out. It should be noted, that most images in this set display an inner-city street scene, so that the use of this set is rather limited.

4.5 Tag Selection

As we intend to label a broad domain of images, we needed to make sure that the image sets showed enough concepts to cover a wide range of topics. In this respect, the Flickr database allows for a greater flexibility, than the pre-annotated Corel set. However, the Flickr tags also proved to be more problematic than those of the professionally labeled set: Many of the tags are subjective (such as `wow`, `myfav`, or `top10`), over-detailed (name of depicted person), or do not describe the contents of the image (name of the author, group or collection). Furthermore, the level of abstraction is far from consistent, with some photos being tagged very detailed, and others very generalized. This is not only apparent in the number of tags used, but also in the words chosen (i.e. `llama` vs. `animal`, or `manhattan` vs. `city`). Frequent subjective tags were therefore filtered, and common over-specific tags generalized. However due to the large number of unique

[3] http://labelme.csail.mit.edu/
[4] http://labelme.csail.mit.edu/guidelines.html

tags (33,967), such mechanisms could only weaken the problems to some extent, and not eliminate them. Unfortunately, simply filtering the tag set to very frequent tags only, often removed important concepts from images altogether.

Furthermore, the photo sets downloaded from community websites naturally reflect subjects that are popular among the community. These tend to be more personal, than the photo sets found on research websites or in commercial sets. For instance, the most popular Flickr tag (at the time of writing) is *wedding*, which is probably due to the fact that many photos are taken during a wedding ceremony. However, it is very unlikely that such an emphasis on wedding photos applies to other image collections, such as those found on a news site or a blog. For this reason, the concepts used as search terms for image aggregation did not reflect the top tags of the website they were downloaded from. Instead, a subset of the most popular tags was manually selected to reflect a broader range of topics. Furthermore, the actual distribution of photos over these categories was neglected, and a more uniform distribution chosen instead. While these measures helped create a more objective categorization, it should be noted that the photo contents themselves still tend to be much more personalized than those found on general websites. For instance, even though we did not explicitly download wedding photos in some of the photo sets, this keyword was still among the most frequent in all downloaded sets from Flickr.

The fotocommunity set, on the other hand, has a relatively small number of distinct categories, and thus better avoids the problem of over-detailed tags. Also, as the category tree seems to be maintained by a few people only (presumably the site's administrators), the descriptions are clear, objective and consistent. However, unlike the other sets, here every image is described by a single concept only. Photos that depict more than one concept are thus only partially described by their tags.

5 Experiments and Results

In the most basic test, we classify and tag each photo using a FIRE server configured to return the 5 nearest neighbor images using color- and texture-histogram features, and the Jensen-Shannon Divergence. These image lists are converted to a list of tags, and the top 4 tags make up the result. The test results are displayed in Figure 2 (left) for the Corel and Flickr image sets, using leave-one-out classification on both sets. Performance is given by the following measures:

- *Mean Precision*: The mean tag precision, i.e. $\frac{|\{\text{correct tags}\}\cap\{\text{returned tags}\}|}{|\{\text{returned tags}\}|}$ over all images.
- *Mean Recall*: The mean tag recall, i.e. $\frac{|\{\text{correct tags}\}\cap\{\text{returned tags}\}|}{|\{\text{correct tags}\}|}$ over all images.
- > 0 *Precision*: The portion of results, that had a precision greater than zero (i.e. at least one correct tag).

As the results show, in terms of precision and recall, the Flickr database is very competitive to the Corel set. However, the non-zero precision is much lower in the

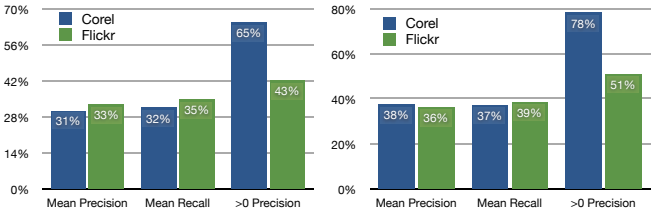


Fig. 2. The tagging results, using color and Tamura texture histograms (left), and weighted histograms (right)

Flickr set, where the majority of images were labeled exclusively with incorrect tags.

Results improve when employing weighted histograms for both the color and texture data. Here, a query tree is used, that merges the result lists of 3 separate FIRE servers - one for each region into one result list. As Figure 2 (right) shows, performance does increase overall, but the non-zero precision gap between the two photo sets remains.

In a slightly more advanced experiment, we adjust the FIRE servers to return 10 nearest neighbors each, and raise the number of tags in the result to 15. The reason for this change, is that such a configuration allows us to analyze the result lists in more detail: By sorting the top tags by their confidence, we can measure the performance of the tagging system for any number of returned tags n ($n \leq 15$), simply by extracting the top n tags only. Also, instead of leave-one-out classification we use disjunct model and test sets for evaluation. Using these parameters and weighted histogram features, we gathered the non-zero precision rates for $5,360$ test images from the Corel dataset. In order to easily match result tags to the true tags in performance measurement, both the test and model sets must use a similar tag dictionary. In this case, we chose a disjunct set of $21,444$ annotated images from the Corel database to act as our model. The results are given in Figure 3. The graph shows, that even when returning just one tag, we obtain a coverage of over 50%. When returning 15 tags, the non-zero precision rate improves to 93%.

The same experiment was repeated for the Flickr set. Here, we downloaded two image sets from the Flickr database, using the same list of search terms (a portion of which is shown in Table 2), but making sure the image sets themselves were disjunct. For every keyword k, a number of images were downloaded, that were tagged with the tag set T_k, and $k \in T_k$. Although this method does not necessarily aggregate images, where k is the main concept, it does guarantee that at least one keyword in each image description occurs in both sets. A total of $26,238$ Flickr images were used in the model set, and $5,247$ images for testing. We tested on a filtered version of the Flickr tags, and on the original tags directly. To filter the Flickr tags, the top tags, with frequencies > 100 were manually reviewed, and those tags removed, that were overly subjective or abstract. As the results in Figures 4 and 5 show, manual tag preprocessing is essential to

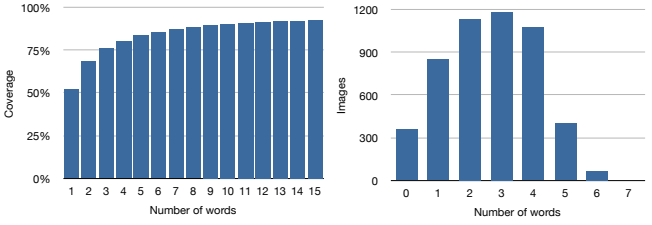


Fig. 3. Tagging performance on a subset of the Corel database (5360 images). The graph on the left shows the percentage of images that are tagged with at least one correct tag, when the top n words are returned. The graph on the right is a histogram of the number of correct words for each image.

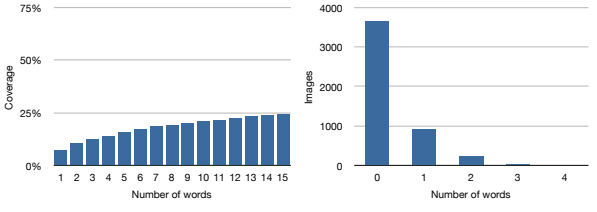


Fig. 4. Using an unfiltered Flickr tag set directly for tag evaluation shows very poor performance

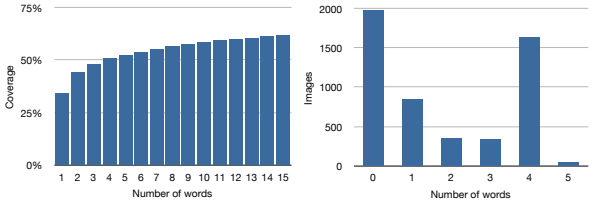


Fig. 5. Tagging performance on the filtered Flickr image set (5247 images). Here, we reach a maximum non-zero precision rate of 62%, and a 34% coverage when returning just one tag.

acceptable performance. However, even after the tags have been filtered, we fall short of reaching the same accuracy as in the Corel test. There are several possible explanations for the poor performance of the Flickr set. One obvious problem is the high diversity of Flickr tags, that very often describe non-relevant image properties (despite having been somewhat pre-filtered). The question is whether a larger model is able to overcome the difficulty of this diversity.

Thus, to analyze whether the employed model is too small for accurate tagging, we set up FIRE servers with various model sizes, ranging from around 100 images up to around 26,000 images for the Corel set, and 50,000 images for the Flickr set. Approximately 500 disjunct images from both sets each were tested against their counterparts. The models for each size were obtained by taking the

full image sets and down-sampling them to the desired sizes. Due to time constraints, we only used color and Tamura histogram features for the evaluation. The performance of each test for the two image sets is shown below in Figure 6. While the recall and precision of the Corel test are asymptotically bounded by approximately 38%, the percentage of images with a non-zero precision shows a less smooth curve with a local maximum at a model size of roughly 13,400 images. These results suggest, that a larger model size (than 26,000 images) would not lead to significantly higher accuracy. The Flickr results, on the other hand, show that even model sizes larger than 25,000 images can still lead to significantly higher scores. Thus it may make sense to employ even larger model databases for the Flickr images than used in our tests.

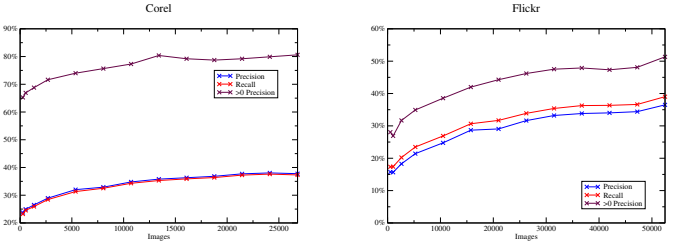


Fig. 6. Approximately 500 images were tested from the Corel and Flickr image set, using FIRE servers with increasing model sizes

Finally, we analyze how the number of images that are returned from the FIRE server affects performance on both sets. This should not be confused with the number of *tags* that make up the result. Recall that each result image from the FIRE server is mapped to a set of tags, where each tag is given the score of the image. Identical tags are merged into one by summing their scores, and the top n tags are extracted to form the result. The more images the FIRE server returns, the more tag sets contribute to the final result list. However, image results that are so distant from the query image, that their score is close to zero, may not be of any relevance to the result at all. By configuring our FIRE servers to return the top k images for various $k \leq 100$, we obtain the results shown in Figure 7. Note that the initial drop of precision is due to the fact that a very low k results in only a small number of returned tags (usually ≤ 4). As we chose to return the top 10 tags in each test, this requires at least $k = 3$ returned images from the FIRE servers. The Corel results show, that roughly $k = 30$ returned images gives an optimal result. The Flickr results on the other hand show a continuing drop in precision, and suggest that a much smaller k leads to overall better results. The explanation for this phenomenon is most likely that popular tags outweigh the correct tags for large k. Figure 8 supports this assumption, showing the percentage of results that contain the popular tag `wedding` for each k. As the result shows, for large k the popular keyword occurs more and more frequently, to the point where over 20% of the images are marked with `wedding`.

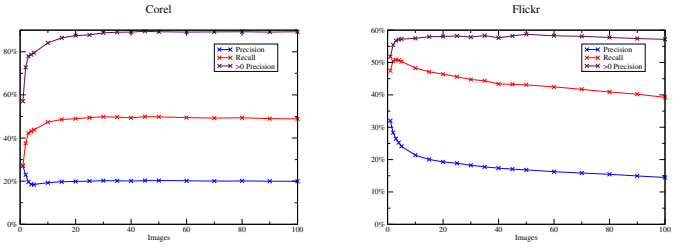


Fig. 7. Again, roughly 500 images were tested from the Corel and Flickr image set, using FIRE servers returning an increasing amount of images

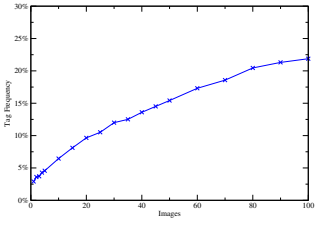


Fig. 8. The frequency of images that were tagged with `wedding` increases with the number of images returned by the FIRE server

Of course, this is far from the actual proportion of our images that depict a wedding ($\approx 1\%$).

To overcome the problem of over-frequent tags, another large Flickr set was downloaded, and repeatedly subsampled, until the tag frequency variance was sufficiently small. However, despite solving the problem of popular tags, overall performance increased only minimally.

At this point, it makes sense to ask whether automatic performance evaluation is suitable for the Flickr image set at all. Simple keyword matching of the result to the true data may simply be too strict for such a large number of distinct tags. Therefore, we manually evaluated the results returned by Tagr. Two configurations of the system were used: One, trained on the Corel set, and another on the Flickr set. A test set of 1000 random Flickr images were sent to each system, and the tag lists returned by the tagging system were manually reviewed. Those tags that clearly described some concept of the image were marked as correct. Figure 9 shows the performance of the Flickr test set on the Corel and Flickr based tagging systems. The results show, that while the Corel system showed performance consistent to the one measured using automatic evaluation, the results for the Flickr model improved greatly. This suggests that Flickr based tagging does indeed find fitting descriptive words for many images. However these descriptions do not match the tags of the ground truth, submitted by the community users. In fact, during this brief evaluation, a number of critical issues were observed:

– The high diversity and ambiguity of Flickr tags became quickly apparent. For instance, images of people are often tagged with `friends`, rather than `people`. Filtering these subjective words out, removes the concept of *people* from the description altogether. On the other hand, replacing occurrences of `friends` with `people` leads to incorrect descriptions for the many animal photos labeled with this tag.
– Occasionally, users annotate whole image sets with the same set of tags. However, usually these tags only apply to a subset of the collection.
– Some of the Flickr photos are greatly distorted or stylized, making an accurate tagging extremely difficult.

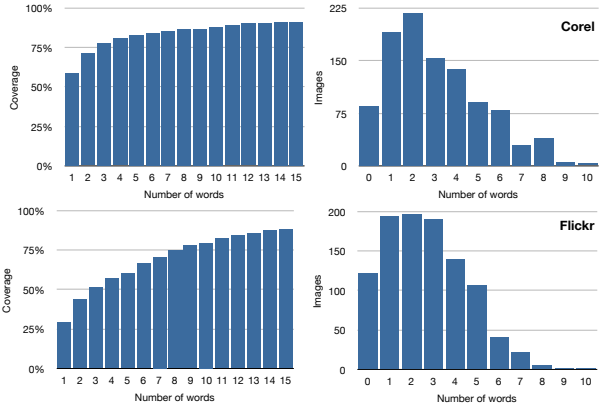


Fig. 9. Roughly 1000 images from the Flickr set were manually evaluated against two models. Tagging performance increased greatly on the Flickr model (bottom), when compared to the results of automatic performance evaluation. Still, the Corel set (top) shows the overall better results.

Using the Corel image set as a model for tagging still provided the overall better results. However, it should be noted, that the Corel tags tended to be much more general than the Flickr tags. Many concepts were missing in the Corel set altogether. High scores were still obtained due to the many general tags, such as `people`, `ground` and `wall`, that occured in 99.3% of the results returned by the Corel based system, and which applied to the majority of images. On the other hand, users on Flickr rarely tag an image with `wall`, even if a wall is depicted, and instead tend to focus on the specifics of the image subject. In this respect, the Flickr set provides an interesting alternative, if more detailed or specialized tags are important. However, the poor performance, most notable when returning less than 4 tags, must be kept in mind.

Overall, these results show some of the challenges involved with using photo sets from community driven web sites. In order to evaluate whether photo sets from other websites show similar results, the (weighted) histogram methods were additionally tested on the remaining sets. Figure 10 shows the results obtained

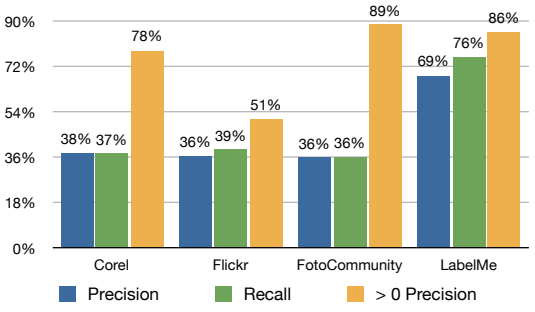


Fig. 10. The precision, recall and non-zero precision for all of the used photo sets

for all of the photo sets, using leave-one-out classification, and 4 result tags. Recall that the FotoCommunity set is not actually tagged. Instead, we used the hierarchical categorization as tags, so that each image is actually tagged with keywords of various levels of abstraction. For instance, as the highest level can only be either `people`, `nature`, or `scenes`, every annotation list contains exactly one of these instances. Note, that this is not necessarily the case for the result list returned by Tagr. These very frequent tags explain the high non-zero precision rate for the FotoCommunity test. Furthermore, as the keywords describe concepts on a much more general level, than for our earlier photo sets, we chose to use simple histogram features instead of weighted histograms for the FotoCommunity set.

The LabelMe set surprises with its very high accuracy. This is most likely due to the small amount of concepts, and therefore keywords (116) in the set. Nevertheless, as these keywords in the LabelMe set are actually descriptions of certain objects within the images, our scene based tagging system performs remarkably well. Overall these results only give a brief glimpse of the performance of other online image sets. However, they do show which image set criteria may lead to improved tagging performance. Like the Corel database, the FotoCommunity set achieves high non-zero precision by including high-level concepts in its list of keywords. The LabelMe set shows that for small image domains, online databases, used as annotation models, can produce very satisfying tagging results.

6 Conclusion

We have shown that it is possible to utilize the internet and its ever-growing community-driven image databases to obtain large annotated image sets, that may be used for automatic image tagging of a broad image domain. Our analysis shows that the direct use of these image sets, without any further filtering or other processing, does not provide satisfying results. Critical issues, such as subjective or non-relevant keyword descriptions, greatly diminish the overall quality of the concept descriptions, and in turn lead to poor results of the tagging system. However, if appropriate measures, such as keyword filtering or uniform

keyword sampling, are employed to overcome these drawbacks, the resulting tagging system may in fact be more suitable to an annotation task, than a commercial set. The reason for this is that the immense size of online databases, such as Flickr, allow a much more flexible aggregation of concepts, that can be tuned to the intended domain. In our case, we were able to aggregate a much higher quantity of concepts from online databases, than from the Corel set, which was previously annotated with a fixed set of tags from a fixed domain. Measuring performance on such a vast set of image tags is challenging, and in most cases will require tedious manual evaluation of the tagging results. Even if the tagging system was trained on a carefully selected annotated image set, random test images are most likely tagged with a different set of keywords than those found in the model set. Leave-one-out classification or cross-validation methods may help here, but have the downside of operating exclusively on the preselected imagery, which may differ greatly from the one found in an installed environment. Furthermore, the keyword preprocessing itself may introduce new problems to the tag set. For instance, although filtering unwanted words from the tag lists may be beneficial to some images, in others it may remove important concepts from the description. Ambiguities in such subjective descriptions make a simple replacement of these tags by more appropriate ones difficult.

Although our analysis highlights many important aspects of image tagging using community image databases, this work is far from complete. Many important questions have yet to be answered. For instance, how do the results change when focussing on only a small domain ($2-5$ different keywords)? How well do community-driven image sets work on classifying regions of images? How useful are they for classification methods other than nearest-neighbor search? Can performance be increased by incorporating ontologies into the tagging process as proposed in [8]? While these questions open new areas of research, the most important next steps will be the continuing analysis of current results. Most importantly, more user studies on standard data sets must be performed, to evaluate real-world performance of various configurations of the tagging system.

References

1. Barnard, K., Duygulu, P., Forsyth, D., de Freitas, N., Blei, D.M., Jordan, M.I.: Matching words and pictures. Journal of Machine Learning Research 3, 1107–1135 (2003)
2. Barnard, K., Forsyth, D.: Learning the semantics of words and pictures. In: Eighth International Conference on Computer Vision, vol. 2, p. 408 (2001)
3. Birchfeld, S.T., Rangarajan, S.: Spatiograms versus histograms for region-based tracking. In: IEEE Conference on Computer Vision and Pattern Recognition, June 2005, vol. 2, pp. 1158–1163 (2005)
4. Deselaers, T., Keysers, D., Ney, H.: Fire - flexible image retrieval engine. In: Peters, C., Clough, P., Gonzalo, J., Jones, G.J.F., Kluck, M., Magnini, B. (eds.) CLEF 2004. LNCS, vol. 3491, pp. 688–698. Springer, Heidelberg (2005)
5. Jeon, J., Lavrenko, V., Manmatha, R.: Automatic image annotation and retrieval using cross-media relevance models. In: Proceedings of the 26th Intl. ACM SIGIR Conference, vol. 3, pp. 119–126 (2003)

6. Li, J., Wang, J.Z.: Real-time computerized annotation of pictures. In: Proceedings of the 14th annual ACM international conference on Multimedia, pp. 911–920 (2006)
7. Mori, Y., Takahashi, H., Oka, R.: Image-to-word transformation based on dividing and vector quantizing images with words. In: First International Workshop on Multimedia Intelligent Storage and Retrieval Management (1999)
8. Srikanth, M., Varner, J., Bowden, M., Moldovan, D.: Exploiting ontologies for automatic image annotation. In: Proceedings of the 28th annual international ACM SIGIR Conference, pp. 552–558 (2005)
9. Stricker, M., Dimai, A.: Color indexing with weak spatial constraints. In: Proceedings of SPIE, March 1996, vol. 2670, pp. 29–40 (1996)
10. Tamura, H., Mori, S., Yamawaki, T.: Textural features corresponding to visual perception. IEEE Trans. Systems, Man, and Cybernetics 8, 460–472 (1978)
11. Wenyin, L., Dumais, S., Sun, Y., Zhang, H., Czerwinski, M., Field, B.: Semi-automatic image annotation 01, 326–333 (2001)
12. Yavlinsky, A., Schofield, E., Ruger, S.: Automated image annotation using global features and robust nonparametric density estimation. In: Leow, W.-K., Lew, M., Chua, T.-S., Ma, W.-Y., Chaisorn, L., Bakker, E.M. (eds.) CIVR 2005. LNCS, vol. 3568, pp. 507–517. Springer, Heidelberg (2005)

Geo-temporal Structuring of a Personal Image Database with Two-Level Variational-Bayes Mixture Estimation

Pierrick Bruneau[1,2], Antoine Pigeau[1],
Marc Gelgon[1,2], and Fabien Picarougne[1]

[1] Nantes university, LINA (UMR CNRS 6241), Polytech'Nantes
rue C.Pauc, La Chantrerie, 44306 Nantes cedex 3, France
[2] INRIA/IRISA Atlas project-team
firstname.surname@univ-nantes.fr

Abstract. This paper addresses unsupervised hierarchical classification of personal documents tagged with time and geolocation stamps. The target application is browsing among these documents. A first partition of the data is built, based on geo-temporal measurement. The events found are then grouped according to geolocation. This is carried out through fitting a two-level hierarchy of mixture models to the data. Both mixtures are estimated in a Bayesian setting, with a variational procedure: the classical VBEM algorithm is applied for the finer level, while a new variational-Bayes-EM algorithm is introduced to search for suitable groups of mixture components from the finer level. Experimental results are reported on artificial and real data.

1 Introduction

This paper addresses the goal of automated structuring a collection of geo-temporally annotated documents. A typical motivation arises from sets of images that can be captured and annotated automatically from mobile camera phones. Schemes for navigating efficiently into a shoebox of personal photographs have attracted growing attention in the past few years, as reviewed in the next section, using image content or meta-data provided by the camera, or a combination thereof. Our proposal does not consider the image content.

Geolocation and time are naturally essential criteria for this organization process [1], as they are both quite reliably measured and provide valuable navigation axes into one's memory and onto man-machine user interfaces. To address large amounts of data, we aim at determining data groups (i.e. clustering) into this geo-temporal space, in order to recover meaningful *events* and *places*. Let an *event* be a group of images that are close both in time and space, while a *place* is a group of data consistent from the sole geolocation criterion. The purpose of this task is to enable efficient browsing through time at a higher level of granularity than data, and hyperlinking co-located events that are distant in time.

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 127–139, 2010.

A main point of the paper is the way these two clustering operations are carried out and relate to one another. First, at the finer level, events are clusters directly identified from image meta-data in three-dimensional space (t, x, y). Then, at the coarser level, places are identified by forming groups of the previously found events, but considering only the location features (i.e. discarding time).

The technical framework for formalizing and solving this issue is that of Gaussian Mixture models (GMM) [2], as classically employed for numerous clustering tasks [3,4,5,6,7], including geo-temporal data [8]. For the problem at hand, where a hierarchy is sought with as little supervision as possible, we propose an original technique, based on the variational approach to Bayesian estimation at each level. The Variational Bayes framework has shown an efficient way of addressing mixture model estimation [9]: while Bayesian modelling enables regularization of estimates (avoiding degenerate situations for covariance matrices) and provides a principled manner of determining adequate model complexity (practically, number of clusters), the associated estimation algorithm often known as VBEM preserves the general form and the relatively low complexity of the Expectation-Maximization (EM) algorithm, but further handles the model complexity issue efficiently.

We propose to extend VBEM to handling a hierarchy of models: we describe in this paper how, also at the coarser level, mixture parameters for places may be inferred from a variational-Bayes EM algorithm applied to event-level components parameters. In other words, two slightly different variational-Bayes mixture estimation procedures are applied: the classical one, operating on punctual geo-temporal data, then the novel version, operating on component-level geolocation-only mixture parameters. The abovementionned advantages of VBEM algorithms are thus generalized to this coarser layer. Besides, because of the hierarchical constraint, the associations between events and places are explicit, in contrast with what would be obtained by determining both of them from the data. At the same time, thanks to the probabilistic framework, uncertainties on these associations are preserved, enabling various decision-making policies. Finally, because this coarse-level algorithm operates on mixture parameters rather than data, it only requires moderate computation cost.

Closely related work offering iterative component-grouping solution for building hierarchies on mixture models were proposed in [10] and [11]. However, they address a maximum likelihood estimate (or similarly, minimizing Kullback-Leibler loss) rather than a Bayesian setting, and hence leave open the issue of model complexity.

Let us mention that the nature of the scheme is also straightforwardly amenable to extensions: the data may be processed incrementally with an initialization/update mechanism in the probabilistic assignments at component-level, and events provided by multiple users may be handled, thus identifying common events, common places. We do not cover these perspectives herein.

From the user point of view, the main advantage of the obtained hierarchy is to facilitate the browsing task, a point emphasized on a mobile device. Indeed

such a device presents energy and interface constraints (for example a small screen and poor input keys) that raise the needs for adapted tools to browse the collection. Our hierarchical aspect improves this task since the coarse-level provides a summarization of the collection events. It enables then to decrease user interactions to finding a specific image and provides energy saving since each summary can be represented with a pertinent image subset.

The remainder of this paper is organized as follows. Section 2 reviews work pertaining to organization of personal images. We then disclose the proposal based on a hierarchy of mixture models (section 3). Experimental results are then provided in section 4, and conclusions and perspectives are drawn (section 5).

2 Related Work

Time stamp is the obvious criterion for ordering pictures, as well as for identifying groups of images close in time (i.e. assumed to have a common topic). Segmenting the sequence of time stamps has been viewed in [12,13] as the incremental detection of gaps, with the advantages that data may be processed as a flow, and each temporal unit is not assumed to be generated from a particular parametric law. However, there result a somewhat arbitrary definition of what is a significant "gap" in time.

Most works on personal image indexing now cope with image location. Practically, handy GPS systems are now largely popular to provide this information (we disregard herein all GPS measurement problems). Systems such as WWMX [14] or Flickr propose a map-based interface to browse the collection. The main problem of such approaches is that the map gets cluttered, when the number of images grow, especially on handheld devices. Alleviating this issues is indeed a main motivation for the work in the present paper. In this direction, building compact representations of the images set and easy navigation procedures has been proposed in [14], where images are aggregated in accordance with the map scale, while [15] selects relevant images from multi user collection based on their meta-data (see below).

Directly combination of the temporal and geographical meta-data, which is the focus of the present paper, was put forward in [16,8], which also organize an image collection hierarchically, based on time and location clusters. A series of heuristic rules derived from user's expectations are implemented [16] to build a geo-temporal hierarchy of events. In [8], we proposed an incremental EM algorithm to carry out distinct temporal and spatial hierarchical classifications, significantly different from the present paper with regard to the relation between geolocation and time, the way the hierarchy is built, and the technique for conducting Bayesian estimation. Finally, time and geographical structuring can also be combined with image features [17], or with the camera settings [18]. Such criteria may indeed be of interest if applied on subset of images corresponding to an event.

Recent contributions extend these principles to the multi user context [15,19,20,21]. Popular websites as Flicker or GoogleMap enable users to share

their personal collections, leading to potentially huge amounts of images. Experiments on user with the Zurfer system [19] show that favorite organization criteria differ from the single user context: users prefer to browse image sets according to social interactions (photos from friends and family members). Let us however notice the distinction between pictures authored by these relatives and personal pictures involving theses persons (such as present in the picture). This was for instance taken into account in [15], where the distance, in a social network, between the image authors and the query author, is taken into account to select a representative image. The importance of images and image cluster is assessed through a heuristic combination of textual tag originality (tf/idf), the diversity of image authors in the location of interest, and various other criteria. This work was extended in [19] to include image content. In [20], a temporal variable is studied, that reflects the density of pictures (from multiple users) on the temporal axis, enabling temporal determination of events. Interestingly, it compares to our proposal in the same way Parzen density estimation compares to mixture models density estimators. Finally, let us notice that, in such multi-user search scenarios, image browsing is generally restricted to a subset of the complete collective collection (generally a specific location).

3 Determinating Events and Places with Hierarchical Variational-Bayes

3.1 Clustering into Events with VBEM on Punctual Geo-temporal Data

In this section, we aim at clustering images based on their metadata in three-dimensional (x,y,t) space. Briefly stated, we model the data as sampled from a Gaussian mixture, and carry out a Bayesian estimation of model parameters by means of a variational approximation.

A Gaussian mixture is defined by the following probability distribution function (pdf):

$$p(x) = \sum_{k=1}^{K} \omega_k \mathcal{N}(x \mid \mu_k, \Lambda_k^{-1}) \tag{1}$$

where x is a d-dimensional feature vector and $\mathcal{N}(. \mid \mu_k, \Lambda_k^{-1})$ is a Gaussian pdf with mean vector μ_k and precision matrix Λ_k. In the remainder of this paper, we will designate $\mathcal{N}(. \mid \mu_k, \Lambda_k^{-1})$ as the k-th component of the GM. $\Omega = \{\omega_k\}$ is a weight vector associated to the components, following the constraint $\omega_k \geq 0 \; \forall k, \sum \omega_k = 1$. We introduce a lightweight notation for the GMM parameters: $\theta = \{\Omega, \mu, \Lambda\}$ where $\mu = \{\mu_k\}$ and $\Lambda = \{\Lambda_k\}$.

In our case, where K is unknown, Maximum Likelihood Estimation (MLE) is not applicable to determine jointly K with model parameters. Although this issue may be overcome using penalized likelihood criteria to compare models of various complexities (AIC [22], BIC [23]), this requires computation of each model separately before comparison. Instead of treating model parameters as

unknown scalars from which we seek MLE, we can define pdfs over these parameters, leading to a fully Bayesian approach. In this context, we can define a variational distribution that, in its optimal setting, will approximate the true posterior distribution in the KL sense [9,2]. Furthermore, using exponential distributions as Gaussians or multinomials allows to define prior pdfs over parameters. Therefore, by choosing an appropriate prior, estimating a correct K will now be part of the global estimation process. Indeed, under this condition the estimation will typically lead to a model with some insignificant component weights that can be pruned ([2]). Hence, by choosing an initial K sufficiently large, we will automatically obtain an effective K (let us denote it K'). The restriction of variational distributions to factorized versions allows to decline coupled update equations over latent variables and parameters. As there is no closed-form solution for this equations system, we can iteratively find a local MAP (Maximum A Posteriori) pdf by using an EM-based algorithm [9,2], often known as VBEM. Besides handling model complexity, setting a spherical prior of covariance matrices helps avoid poor estimates, that classically plague clusters that are responsible for little data.

3.2 Grouping Events into Places with a Component-Level VBEM Algorithm

Once events are identified, we attempt to find groups of events, based on the mere parameters of the mixture that describe this set of events. We now describe the technique employed to this aim.

We follow notations used [2] for punctual data. Classically, mixture variational estimation considers a set of data $X = \begin{pmatrix} x_1^T \\ \dots \\ x_N^T \end{pmatrix}$ and $Z = \begin{pmatrix} z_1^T \\ \dots \\ z_N^T \end{pmatrix}$ that is assumed to be generated from the mixture. x_i is a d-dimensional feature vector and z_i the associated binary variable indicating from which component x_i was generated (e.g. from k-th component $\equiv z_{ik} = 1, z_{ij} = 0\ \forall j \neq k$). In the clustering context, Z is hidden, and the purpose of the procedure is to compute a joint estimate of θ and Z. The associated pdfs are:

$$p(Z \mid \Omega) = \prod_{n=1}^{N} \prod_{k=1}^{K} {\omega_k}^{z_{nk}} \tag{2}$$

$$p(X \mid Z, \mu, \Lambda) = \prod_{n=1}^{N} \prod_{k=1}^{K} \mathcal{N}(x_n, \mu_k, \Lambda_k^{-1})^{z_{nk}} \tag{3}$$

Now consider an arbitrary mixture defining L components, with parameters $\theta' = \{\Omega', \mu', \Lambda'\}$. Typically this model might have redundant components, or might be a group of mixtures. We then assume that X and Z were i.i.d sampled from this distribution. It is therefore possible to regroup X by the component that originated its various items. It leads us to the following formalism: $X = \{\hat{x}_1, \dots, \hat{x}_L\}$ with $\text{card}(X) = N, \hat{x}_l = \{x_i \mid z_{il} = 1\}$ and $\text{card}(\hat{x}_l) = \omega_l' N$. We are

now going to express the distributions (2) and (3) w.r.t this formalism. To achieve tractability, we make the following assumption: $\forall x_i \in \hat{x}_l, z_{ik} = \text{const} = z_{lk}$. Thus we can rewrite the expression (3):

$$p(X \mid Z, \mu, \Lambda) = \prod_{k=1}^{K} \prod_{l=1}^{L} p(\hat{x}_l \mid Z, \mu_k, \Lambda_k)^{z_{lk}} \tag{4}$$

$$p(X \mid Z, \mu, \Lambda) = \prod_{k=1}^{K} \prod_{l=1}^{L} \left[\prod_{i=1}^{\omega_l' N} \mathcal{N}(x_{li} \mid \mu_k, \Lambda_k^{-1}) \right]^{z_{lk}} \tag{5}$$

$$\ln p(X \mid Z, \mu, \Lambda) = \sum_{k=1}^{K} \sum_{l=1}^{L} z_{lk} \left[\sum_{i=1}^{\omega_l' N} \ln \mathcal{N}(x_{li} \mid \mu_k, \Lambda_k^{-1}) \right] \tag{6}$$

For N sufficiently large, we can make the following approximation:

$$\sum_{i=1}^{\omega_l' N} \ln \mathcal{N}(x_{li} \mid \mu_k, \Lambda_k^{-1}) \simeq \omega_l' N E_{\mu_l', \Lambda_l'} \left[\ln \mathcal{N}(x \mid \mu_k, \Lambda_k^{-1}) \right] \tag{7}$$

This statement is known as *virtual sampling*, and was introduced in [24,10] in the context of max. likelihood estimation.

We can also write:

$$E_{\mu_l', \Lambda_l'} \left[\ln \mathcal{N}(x \mid \mu_k, \Lambda_k^{-1}) \right] = \int \mathcal{N}(x \mid \mu_l', \Lambda_l'^{-1}) \ln \mathcal{N}(x \mid \mu_k, \Lambda_k^{-1}) \, \mathrm{dx} \tag{8}$$

$$\begin{aligned} E_{\mu_l', \Lambda_l'} \left[\ln \mathcal{N}(x \mid \mu_k, \Lambda_k^{-1}) \right] = -KL \left(\mathcal{N}(x \mid \mu_l', \Lambda_l'^{-1}) \parallel \mathcal{N}(x \mid \mu_k, \Lambda_k^{-1}) \right) \\ - H(\mathcal{N}(x \mid \mu_l', \Lambda_l'^{-1})) \end{aligned} \tag{9}$$

with $KL(q_0 \parallel q_1)$ the KL divergence of q_1 from q_0 and $H(q_0)$ the entropy of q_0. These two terms have closed-form expressions [25]. Thus by reinjecting (9) into (7), and then (7) into (6), we obtain the following expression:

$$\begin{aligned} &\ln p(X \mid Z, \mu, \Lambda) = N \sum_{k=1}^{K} \sum_{l=1}^{L} z_{lk} \omega_l' \\ &\left[-KL \left(\mathcal{N}(x \mid \mu_l', \Lambda_l'^{-1}) \parallel \mathcal{N}(x \mid \mu_k, \Lambda_k^{-1}) \right) - H(\mathcal{N}(x \mid \mu_l', \Lambda_l'^{-1})) \right] \end{aligned} \tag{10}$$

$$\begin{aligned} &\ln p(X \mid Z, \mu, \Lambda) = N \sum_{k=1}^{K} \sum_{l=1}^{L} z_{lk} \omega_l' \\ &\left[\frac{1}{2} \ln \det \Lambda_k - \frac{1}{2} \mathrm{Tr}(\Lambda_k \Lambda_l'^{-1}) - \frac{1}{2} (\mu_l' - \mu_k)^T \Lambda_k (\mu_l' - \mu_k) - \frac{d}{2} \ln(2\pi) \right] \end{aligned} \tag{11}$$

The formalism change we made also has consequences on (2): as we previously stated that $z_{lk} = z_{nk} \; \forall x_n \in \hat{x}_l$, we can write:

$$p(Z \mid \Omega) = \prod_{n=1}^{N} \prod_{k=1}^{K} \omega_k^{z_{nk}} = \prod_{l=1}^{L} \prod_{k=1}^{K} \omega_k^{N \omega_l' z_{lk}} \tag{12}$$

Variational update equations are partially based on moments evaluated w.r.t $p(Z)$ and $p(X)$. Therefore we can review the cascading consequences relatively to the method introduced in [2].

$q(Z)$ expression is based on the computation of the $\ln(\rho_{nk})$ terms. As a consequence of (11) and (12), it now reduces to the computation of:

$$\begin{aligned}\ln(\rho_{lk}) = \frac{N\omega_l'}{2}\left(2E[\ln \omega_k] + E[\ln\det \Lambda_k] - d\ln(2\pi)\right)\\ -\frac{N\omega_l'}{2}\left(E_{\mu_k,\Lambda_k}\left[\mathrm{Tr}(\Lambda_k \Lambda_l^{'-1}) + (\mu_l' - \mu_k)^T \Lambda_k(\mu_l' - \mu_k)\right]\right)\end{aligned} \tag{13}$$

This leads to the computation of the set $\{r_{lk}\}$ and defines $q(Z)$ in its optimal setting.

The moment w.r.t μ_k and Λ_k is easily evaluated to give

$$\frac{d}{\beta}_k + \nu_k\left[\mathrm{Tr}(W_k \Lambda_l^{'-1}) + (\mu_l' - m_k)^T W_k(\mu_l' - m_k)\right].$$

The pdfs $q(\Omega)$ and $q(\mu, \Lambda)$ are also modified. In their optimal setting, the update equations become:

$$\alpha_k = \alpha_0 + \sum_l N\omega_l' r_{lk} \tag{14}$$

$$\beta_k = \beta_0 + \sum_l N\beta_l' r_{lk} \tag{15}$$

$$m_k = \frac{1}{\beta_k}\left(\beta_0 m_0 + \sum_l N\omega_l' r_{lk}\mu_l'\right) \tag{16}$$

$$W_k^{-1} = W_0^{-1} + \beta_0 m_0 m_0^T - \beta_k m_k m_k^T + \sum_l N\omega_l' r_{lk}(\mu_l'\mu_l^{'T} + \Lambda_l^{'-1}) \tag{17}$$

$$\nu_k = \nu_0 + \sum_l N\omega_l' r_{lk} \tag{18}$$

Cycling through these update equations implements an EM-based algorithm, analogously as presented in [2,9].

Also in [2], we learn that the previously described algorithm monotonically decreases the KL distance between the variational pdf and the true posterior. This is equivalent to maximising the lower bound of the complete likelihood. As we can compute this lower bound, and as this bound should never decrease, we can test for convergence by comparing two successive values of the bound. Only terms of the bound that depend on Z or X are impacted, we list these changes below:

$$\begin{aligned}E[\ln p(X \mid Z, \mu, \Lambda)] = \frac{1}{2}\sum_k\sum_l N\omega_l' r_{lk}\\ \left[\ln\tilde{\Lambda}_k - \frac{d}{\beta_k} - \nu_k[\mathrm{Tr}(W_k\Lambda_l^{'-1}) + (\mu_l' - m_k)^T W_k(\mu_l' - m_k)]\right]\end{aligned} \tag{19}$$

$$E[\ln p(Z \mid \Omega)] = \sum_l\sum_k N\omega_l' r_{lk}\ln\tilde{\omega}_k \tag{20}$$

Choosing appropriate priors can be problematic. If we have expert knowledge on the clustering structure to build, this can be used to build it, but we must be able to carry out the problem in a totally unsupervised fashion. Fortunately, it is possible to define uninformative but efficient priors. We set:

1. Uniformly chosen prior means (on the observed data space),
2. Isotropic prior covariances (diagonal values scaled to half of the observed variance)

This approach is valuable if we choose an enough big number of prior components. Doing this enables a sufficiently large search through parameter space, therefore avoiding the worst local optima. Experimentally, we have found that for low dimensional spaces (2 or 3), using 30 to 50 components is a rather good compromise to capture most of the clustering structure while keeping a low computational cost.

4 Experimental Results

We present here results obtained on the proposed method (let's name it VBMerge), first on an artificial, then on a real data set.

We want to evaluate the overall quality of estimations made with VBMerge. For this, we will compare the relative quality of models estimated with VBMerge with respect to a batch VBEM approach. Indeed, the reduction of a too rich model should not degenerate the estimation, in other words it should converge to an estimation close to the true distribution, while reducing significantly the number of components. For the comparison we will use 2 synthetic data sets (sampled from a 4-component 2-dimensional GM with randomly placed means, various covariance matrices and various populations per component) and the *Glass Identification* data set [26]. This data set describes 214 samples of glass with 9 physical (numerical) characteristics. Each sample is associated with a true label from 7 possible. We consider here that the partition of items w.r.t. labels constitutes the distribution we try to discover.

Then we will illustrate the joint usage of VBEM and VBMerge on the suggested geo-temporal context. This data set was introduced in [8] and describes 721 images over spatial and temporal attributes. It was obtained from a personal collection taken in several countries over 3 years. In [8] this data set was used to build a Gaussian mixture hierarchy designed to navigate conveniently in the obtained events. We shall see that VBMerge can be a promising building block in such a context.

4.1 VBMerge vs. VBEM

For the quality assessment, we used the following settings:

– For the synthetic data sets, VBEM is initialised randomly with $K = 30$ (a usually good value for 2D spaces). To generate a redundant GM we use a classic EM algorithm estimation ([27], [2] chap. 9) with $K = 8$.

Indeed, classic EM usually tends to overfit data, so this will be a good point if VBMerge is able to compensate this drawback while giving a good estimate.
– For the *glass* data set, as its dimensionality is much higher we use $K = 300$ for VBEM, as this is necessary to cover the data space properly. For classic EM we use $K = 30$,

For all data sets, and for each setting (VBEM or classic EM-VBMerge), we will measure:

– the empirical cross-entropy (the data set is considered as a sample from the true distribution) of the estimated model w.r.t the true (unknown) distribution.
– the KL divergence of the estimated model w.r.t the true distribution. Of course this calculation is possible only in the case of a synthetic data set, where the true distribution is known. An approximate value of this measure is obtained through a simple sampling scheme (see [2] chap. 11) (100000 draws)
– the number of components in the estimated model
– the couple error [28] measured between the inferred labels (i.e. which component is associated with each datum) and the true labels. This error measures if two items that are in the same (respectively a different) class w.r.t the original distribution also are in the same (resp. different) class in the estimated distribution.
– We made 20 measurements per study case.

Fig 1 displays the mean of these results, with associated standard deviation in brackets. As the variational estimation integrates important characteristics such as a low and appropriate number of components and a good separation between low-entropic components (for a discussion about these aspects, see [8]), we believe that the cross-entropic and KL values are significant quality measurements.

Obtained values on synthetic data set show that there is no significant difference between our VBMerge and the VBEM. Therefore our proposed algorithm applied on a redundant model can lead to a result as good as obtained with an

data \measure	VBEM cross-ent.	VBEM KL div.	VBEM nb. comp.	VBEM couple
synth1	790.3 [2.2]	0.207 [0.011]	5.4 [0.5]	0.19 [0.01]
synth2	781.1 [0.49]	0.054 [0.003]	3 [0]	0.14 [0.06]
GLASS	-366.7 [203.5]		4.0 [0.67]	0.38 [0.08]
data \measure	VBMerge cross-e.	VBMerge KL	VBMerge nb. comp.	VBMerge couple
synth1	817.9 [5.7]	0.118 [0.013]	4.5 [0.97]	0.15 [0.04]
synth2	794.0 [2.1]	0.095 [0.008]	3.7 [0.82]	0.16 [0.02]
GLASS	578.4 [41.2]		2 [0]	0.69 [0.08]

Fig. 1. Benchmarking results

usual variational estimate. On the other hand, the results are much worse on the *glass* data set. This must be due to a bad initial estimate by the classic EM algorithm. Indeed, this data set is not designed for Gaussian mixture modelling at all (e.g. contains some very low entropic variables). A classic EM approach is very sensitive to this problem, and will typically lead to a very degenerated optimum, as VBEM alone is much more robust to such issues [2]).

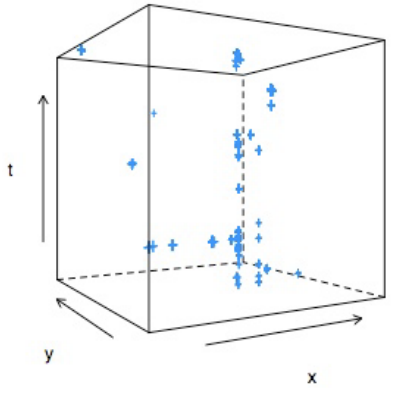


Fig. 2. 3D representation of the image collection (spatial and temporal coordinates). Each + represents a coordinate (t, x, y) of one image. Here Z is the temporal axis.

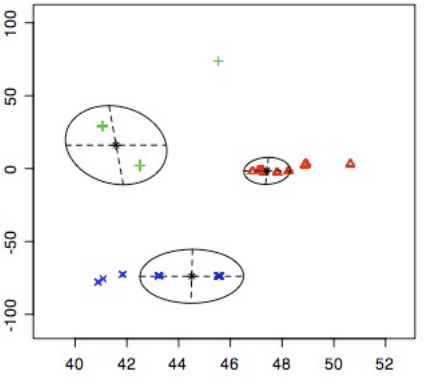


Fig. 3. Obtained 2D Reduced model. The ellipses are the variance of the components. The different symbols represent the data-to-class assignments. The 8 obtained clusters in the *finer* level are here summarized with 3 clusters.

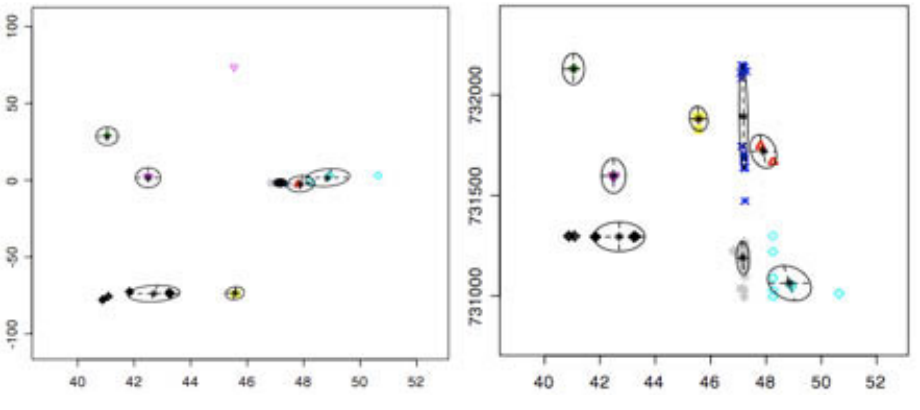


Fig. 4. 3D VBEM inferred model (xy and xz projections). The ellipses are the variance of the components. The different symbols represent the data-to-class assignments. We clearly identify some captured *events* and *places*.

4.2 Building a Mixture Hierarchy on Geo-temporal Data

The data set used for our experiment is represented in figure 2. We will follow these steps to build a simple hierarchy:

1. Estimate a model on geo-temporal data using VBEM (see figure 4),
2. Project this model on the (x, y) sub-space,
3. Reduce this model using VBMerge.

When merging components with VBMerge, we define implicitly a probabilistic mapping of the original components with the merged components. This allows to define a 2-level GM hierarchy.

The second level is presented in Figure 4(a). We obtained 8 compact and well defined classes. All the main location of the user collection was correctly retrieved. The first level on Figure 3 displays the reduced model defined only on (x, y) space (clusters of the *finer* also appears). This can be an entry point for user navigation since the obtained summary seems pertinent: it is composed of just 3 clusters, with well defined boundaries. Nevertheless, the image group situated at the coordinate $[45, 75]$ is isolated: as in [8], small image group tends to be grouped with the nearest clusters.

A user can then browse its collection switching between the different obtained classifications: for example, he can first start by browsing our geographical hierarchy to select a specific location, and then switch to the initial 3-D partition, therefore offering a view of closely related places at a same time.

5 Conclusion

This paper discloses a technique for building a hierarchy of mixture models, where the main contribution is a variational approach to Bayesian clustering of components. The main features of the approach is that it possesses advantages of Bayesian modelling, while being computationally very tractable. The technique was motivated by an applicative need arising from indexing image collections, based on geo-temporal metadata, enriching the browsing possibilities within the collection.

Besides considering multi-users and integrating information from geographical information systems, such as work quoted in Section 2, we believe that structure from time-continuous capture of geolocation, rather than from the mere time instants of pictures, should be investigated to gain valuable insight.

References

1. Rodden, K.: How do people manage their digital photographs? In: ACM Conference on Human Factors in Computing Systems, Fort Lauderdale, pp. 409–416 (2003)
2. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York (2006)

3. Blekas, K., Lagaris, I.E.: Split-merge incremental learning (smile) of mixture models. In: de Sá, J.M., Alexandre, L.A., Duch, W., Mandic, D.P. (eds.) ICANN 2007, Part II. LNCS, vol. 4669, pp. 291–300. Springer, Heidelberg (2007)
4. Celeux, G., Govaert, G.: A classification em algorithm for clustering and two stochastic versions. Computational Statistics and Data Analysis (1992)
5. Celeux, G., Govaert, G.: Gaussian parcimonious clustering models. Pattern Recognition 28, 781–793 (1995)
6. Fraley, C., Raftery, A.E.: Mclust: Software for model-based clustering, density estimation and discriminant analysis. Technical report 415, Department of Statistics - University of Washington (2002)
7. Fraley, C., Raftery, A.E.: Model-based clustering, discriminant analysis and density estimation. Journal of the American Statistical Association (2002)
8. Pigeau, A., Gelgon, M.: Building and tracking hierarchical geographical & temporal partitions for image collection management on mobile devices. In: Proceedings of International Conference of ACM Multimedia, Singapore, pp. 141–150 (2005)
9. Attias, H.: A variational bayesian framework for graphical models. In: Advances in Neural Information Processing Systems (2000)
10. Vasconcelos, N.: Image indexing with mixture hierarchies. In: Proceedings of IEEE Conference in Computer Vision and Pattern Recognition (2001)
11. Goldberger, J., Roweis, S.: Hierarchical clustering of a mixture model. In: NIPS (2004)
12. Graham, A., Garcia-Molina, H., Paepcke, A., Winograd, T.: Time as essence for photo browsing through personal digital libraries. In: Proceedings of the ACM Joint Conference on Digital Libraries JCDL, pp. 326–335 (2002)
13. Platt, J.C., Czerwinski, M., Field, B.A.: PhotoTOC: Automatic clustering for browsing personal photographs. Technical Report MSR-TR-2002-17, Microsoft Research (2002)
14. Toyama, K., Logan, R., Roseway, A., Anandan, P.: Geographic location tags on digital images. In: Proceedings of the eleventh ACM international conference on Multimedia, Berkeley, CA, USA, pp. 156–166 (2003)
15. Jaffe, A., Naaman, M., Tassa, T., Davis, M.: Generating summaries and visualization for large collections of geo-referenced photographs. In: Proceedings of the 8th ACM SIGMM International Workshop on Multimedia Information Retrieval, pp. 853–854 (2006)
16. Naaman, M., Song, Y.J., Paepcke, A., Garcia-Molina, H.: Automatic organization for digital photographs with geographic coordinates. In: Proceedings of the ACM/IEEE Conference on Digital libraries (JCDL 2004), pp. 53–62 (2004)
17. Cooper, M., Foote, J., Girgensohn, A., Wilcox, L.: Temporal event clustering for digital photo collections. In: Proceedings of the ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP), vol. 1, pp. 269–288 (2005)
18. Gargi, U., Deng, Y., Tretter, D.R.: Managing and searching personal photo collections. Technical Report HPL-2002-67, HP Laboratories, Palo Alto (2002)
19. Kennedy, L., Naaman, M.: Generating diverse and representative image search results for landmarks. In: Proceedings of The Seventeenth International World Wide Web Conference, WWW 2008 (2008)
20. Nair, R., Reid, N., Davis, M.: Photo loi: Browsing multi-user photo collections. In: Proceedings of International Conference of ACM Multimedia, pp. 222–223 (2005)

21. O'Hare, N., Gurrin, C., Jones, G., Smeaton, A.F.: Combination of content analysis and context features for digital photograph retrieval. In: Proceedings of of the 2nd IEE European Workshop on the Integration of Knowledge, Semantic and Digital Media Technologies, pp. 323–328 (2005)
22. Akaike, H.: A new look at the statistical model identification. IEEE Trans. on Automatic Control AC-19(6) (1974)
23. Schwarz, G.: Estimating the dimension of a model. The Annals of Statistics 6, 461–464 (1978)
24. Vasconcelos, N., Lippman, A.: Learning mixture hierarchies. In: Neural Information Processing Systems (1998)
25. Blahut, R.E.: Principles and Practice of Information Theory. Addison-Wesley, Reading (1987)
26. Evett, I.W., Spiehler, E.J.: Rule induction in forensic science. Ellis Horwood in Expert Systems, Knowledge Based Systems, 152–160 (1989)
27. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. J. Royal Stat. Soc. B(39), 1–38 (1977)
28. Azzag, H.: Classification hiérarchique par des fourmis artificielles: application à la fouille de données et de textes pour le Web. PhD thesis, Ecole Doctorale Santé Sciences et Technologies, Université François Rabelais Tours (2005)

Unsupervised Clustering in Personal Photo Collections

Edoardo Ardizzone[1], Marco La Cascia[1], and Filippo Vella[2]

[1] DINFO - Dipartimento di Ingegneria Informatica,
University of Palermo, Palermo, Italy
{ardizzon,lacascia}@unipa.it
[2] ICAR - Istituto di Calcolo e Reti ad Alte Prestazioni
Italian National Research Council, Palermo, Italy
filippo.vella@pa.icar.cnr.it

Abstract. In this paper we propose a probabilistic approach for the automatic organization of collected pictures aiming at more effective representation in personal photo albums. Images are analyzed and described in two representation spaces, namely, faces and background. Faces are automatically detected, rectified and represented projecting the face itself in a common low dimensional eigenspace. Backgrounds are represented with low-level visual features based on RGB histogram and Gabor filter energy. Face and background information of each image in the collection is automatically organized by mean-shift clustering technique. Given the particular domain of personal photo libraries, where most of the pictures contain faces of a relatively small number of different individuals, clusters tend to be semantically significant beyond containing visually similar data. We report experimental results based on a dataset of about 1000 images where automatic detection and rectification of faces lead to approximately 300 faces. Significance of clustering has been evaluated and results are very encouraging.

1 Introduction

With the widespread diffusion of digital cameras the cost of taking hundreds of digital pictures and storing them on personal computer is quickly approaching zero. People are then encouraged to take more and more pictures but the consequent risk is that they end up with tens of thousand of pictures stored on their PCs that, without a proper organization, become useless. Currently, the main way to search digital photo libraries is by mean of time of shooting and/or keywords given by the user. This modality of access to the library is definitely unsatisfactory, moreover it requires the user manually associates keywords to each picture. This process has been observed to be inadequate since users usually add few keywords for large set of images and, on the other side, keywords tend to be ambiguous. Time of shooting is a much more reliable cue and it is available for free as digital cameras attach a timestamp, in the EXIF data, to each pictures. However its power in term of searching capabilities is quite limited. An ideal

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 140–154, 2010.

system for image browsing should allow an automatic organization of pictures based on the semantic of photos. Our point is that personal photo libraries show peculiar characteristics compared to general image collection, namely the presence of people in most of the images and a relatively small number of different individuals across the whole library that allow to achieve reliable results with automatic approaches[3]. In particular, in personal photo collection the user is mainly interested in *who* is in the picture (usually a relatively small number of different individuals) and *where* and *when* the picture was shot. When the picture was shot is an information that comes for free as all the digital cameras attach a timestamp to the pictures they take. Even though this temporal information is a very useful piece of information [14] we focused in this work on the other aspects while this issue is object of further investigation. *Who* and *where* are the fundamental aspects of photo information and input images can be intrinsecally split in two domains of interest. Faces are extracted from the images and referred to person identity; the remaning part of the image is considered as image context. Known techniques[4] are used to detected and rectify faces from the data set allowing to project all the samples in a common low dimensional *face space*. Images' background are characterized by mean of low-level features based on color and texture characterizing different contexts (*where*). Also for this aspect, as the typical user is interested to a limited number of different contexts, the link between low-level features and context semantic content can be reasonably established.

To automatically organize image data based on faces and background descriptors we use a mean-shift based approach[7]. Organization of data does not need any human intervention as image features are automatically extracted and parameters of the clustering method are automatically determined according to a proposed entropy based figure of merit. The paper is organized as follows: Section 2 describes the known techniques for organization, storage and content based retrieval for personal photo collection; in Section 3 the proposed description of data composing personal album is given. The details of the image processing and analysis are described in 3.1 and in 3.2. The clustering process is presented in 4. Finally in Sections 5 and 6 are given the results of the experiments and the conclusions.

2 Related Works

One of the first personal photo collection browser has been reported by Kang and Shneiderman[15]. The goal of this system was to enable non-technical users of personal photo collection to browse and search efficiently for particular images. The authors proposed a very powerful user interface but implemented very limited CBIR capabilities. Moreover the search was heavily based on manual annotation of the data. As in personal photos the objects of interest are often people Zhang et al. [25] addressed the problem of automated annotation of human faces in family album. CBIR techniques and face recognition are integrated in a probabilistic framework. Based on initial training data models of each person are built and faces in images are often recognized correctly even in presence of some

occlusions. User interaction during the annotation process is also possible to reinforce the classifier. Experimental results on a family album of a few thousands photos showed the effectiveness of the approach. In a subsequent work [26] some of the authors developed a system where the user is allowed to select multiple images and assign them personal names. Then the system tries to propagate names from photograph level to face level exploiting face recognition and CBIR techniques. Abdel-Mottaleb and Chen [2] also studied the use of faces arrangement in photo album browsing and retrieval. In particular they defined a similarity measure based on face arrangement that can be computed automatically and is used to define clusters of photos and finally to browse the collection. A photo management application leveraging face recognition technology has also been proposed by Girgensohn et al.[12]. The authors implemented a user interface that greatly helps the users in face labelling. Other semi-automatic annotation tecniques for personal photo libraries have also been proposed recently [19,17,9]. Other researcher address the problem of personal photo album management in an image clustering framework. For example hierarchical clustering enable the users to navigate up and down the levels to find images. Navigating the collection is also useful in query-by-example systems to find the initial image. In any case the clusters prototype are a compact representation of classes of similar images and then can be used in browsing or searching the library. The efficacy of the clustering approach, as well as any CBIR system, is obviously affected by the goodness of the image features used to describe the images and the similarity metrics defined over these features. As similarity metrics may not reflect semantic similarity between images, sometimes clusters are not semantically homogeneous. Many techniques have been proposed to refine the automatic clustering approach with human intervention to make cluster semantically homogeneous. Several techniques have been proposed for the clustering of images. For example in[16] the authors use color histogram and histogram intersection distance measure to perform hierarchical clustering. Similarly, Chen et al. [6] used global color, texture and edge histogram and the L_1 distance to define an hierarchical browsing environment. In [11] a self-organizing map is used to let the structure of the data emerge and then to browse the collection. In [20] the authors use a clustering strategy based on Markov Model Mediators to improve efficiency and efficacy of retrieval in distributed image databases. Recently, Goldberger et al. [13] proposed a generalized version of the information bottleneck principle where images are clustered to maximally preserve the mutual information between the clusters and image contents. Experimental results using different image representation are also reported. In other cases the presence of faces in an attempt to bridge the gap between visual and semantic content is exploited. For example in [4] face detection is performed on captioned images and clustering is used to associate automatically extracted names to the faces. In [18] the authors detect faces and describe clothes and nearby regions with color histogram. A similarity matrix of a photo collection is then generated according to temporal and content features and hierarchical clustering is performed based on this matrix. Song and Leung [21] aims at clustering the dataset such that each cluster contains images

of a particular individual. They use face and clothing descriptors to contruct an affinity matrix over the identities of individuals and perform clustering using a normalized-cut approach. In [10] a semi automatic photo annotation system based on enhanced spectral clustering is proposed. They use time, global color correlogram for location/event clustering and local facial features and color correlogram from human body area for face clustering. As automatic techniques cannot guarantee that all the faces in a cluster are related to the same individuals or that an individual is not spread across several clusters, the final validation of the clustering is done by hand.

3 Personal Photo Album Indexing

The focal point of the clustering is the representation of each image in a form suited for clustering. An image can be represented in several spaces allowing to capture different aspects of input data. In the proposed system, each image in the collection is represented with features related to the presence of faces in the image and features characterizing the background[3]. A data oriented clustering allows to generate aggregation structures driven by the regularities in the represented data. In the following sections the processing of visual information in the two chosen representation spaces is described. Faces are preprocessed to reduce the variation of the appearance and are mapped in an auto emerging space employing eigenfaces. The information from background is managed representing parts not associated in a vector space representing low-level features.

3.1 Face Representation

Finding faces in general images is a very challenging task due to variations in pose and illumination. Berg et al.[4] analyzed hundred of thousands of images taken from the Internet to detect faces *in the wild*. In a similar way in our approach each image to be archived in the system is searched for faces. Detected faces are then validated and rectified to a canonical pose and size. The face detector we adopted[23] is usually successful in detecting faces in a quite large range of pose, expression and illumination conditions. In some different application, such as detection of faces in video sequences, a SVM based face detector is used instead [1],[22] for the improved capability in face detection when images are blurred. For the detected faces, we try to detect five features per face *left eye extern corner*, *right eye extern corner*, *tip of the nose*, *mouth left corner*, and *mouth right corner* through SVM detectors and, if detection is successful, we estimate an affine transformation to rescale and align the face to canonical position. Fiducial points detectors have been trained with hundreds of positive and negative examples and Radial Basis Function have been employed for training. Test on detection are very good ($\tilde{9}0\%$ of true positive) A final crop to 100×100 pixels brings each face to a common reference system.

Faces where detection process produces features with low level of confidence were rejected. For successfully detected and rectified faces,a reduced dimension

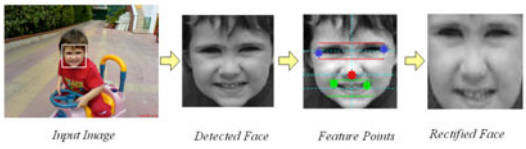


Fig. 1. Example of detected face and corresponding rectified image

face descriptor is computed. The descriptor is a vector $\mathbf{w}$ containing the projection of the rectified and cropped face in a subspace of the global face space. In practice the average face Ψ is subtracted from the 100×100 cropped and rectified face Γ_i and the obtained image Φ is then projected on the eigenspace to obtain $w_i = \mathbf{e}_i^T \Phi$. The face space, as well as the average face, is learned off-line on a significant subset of the image collection and it is not updated. At any time, if most of the faces present in the image collection differ significantly from the training set, it is possible to build a new face space and effortlessly recompute the projection of each detected, rectified and cropped face in the new face space.

3.2 Background Representation

The largest part of semantic information in personal photo is conveyed in areas where faces appear, the remaining part of the image information is attributed the context of the scene. As described above, each picture is processed with the face detector selecting areas containing faces. These areas are approximated with bounding boxes and are dealt as seen in previous section. The remaining part, not representing a face, is then processed as background. Note that the partition of images in multiple areas of interest can be extended using additional detectors (e.g. a detector for entire body ccould be easily integrated in the system). Background information can be represented with a composition of color and texture features. Features are globally evaluated and a single vector for each image is produced. Color information is captured through histograms in the RGB color space. The 60-dimensional global descriptor is computed as the concatenation of the 20-bin histograms of the R, G and B channels. Texture is evaluated through Gabor filters considering 6 different filters, varying 3 orientation and 2 scales. For each filter the energy value is evaluated and represented as a 15 bins histogram. The total feature will be composed by a total of 15x6 components. Since mean values of features in the data space distribution are less indicative, image features are filtered with a sigmoid to stretch the values towards a low or high values according to a parameter α.

4 Image Clustering

4.1 The Mean Shift Algorithm

Mean shift is a technique for kernel density estimation that applies gradient climbing to probability distribution[8]. Given n data points $\mathbf{x}_i, i = 1, 2, ..., n$

in the d-dimensional space R^d, a multivariate kernel density estimator $\hat{f}(\mathbf{x})$ is calculated as

$$\hat{f}(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} K(\frac{\mathbf{x} - \mathbf{x_i}}{h}) \tag{1}$$

where h is the bandwidth and the kernel $K(.)$ is the Epanechnikov kernel defined as:

$$K(x) = \begin{cases} \frac{1}{2V_d}(d+2)(1 - \|\mathbf{x}\|^2) \text{ if } \|\mathbf{x}\|^2 < 1 \\ 0 \qquad\qquad\qquad\qquad \text{otherwise} \end{cases} \tag{2}$$

with V_d representing a volume of a unit d-dimensional sphere. Using a differentiable kernel, the estimate of the gradient density can be written as the gradient of the kernel density estimate(1):

$$\hat{\nabla} f(\mathbf{x}) \equiv \nabla \hat{f}(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} \nabla K(\frac{\mathbf{x} - \mathbf{x}_i}{h}) \tag{3}$$

For the Epanechnikov kernel, shown in (2), the density gradient estimate is:

$$\hat{\nabla} f(\mathbf{x}) = \frac{n_c}{nV_d} \frac{d+2}{h^d} \left(\frac{1}{n_c} \sum_{\mathbf{x} \in S(x)} (\mathbf{x}_c - \mathbf{x}) \right) \tag{4}$$

where $S(\mathbf{x})$ is the hyper-sphere of radius h, having volume $h^d V_d$, centered in $\mathbf{x}$ and containing n_c data points. The quantity $M_h(\mathbf{x})$ defined as

$$M_h(\mathbf{x}) \equiv \frac{1}{n_c} \sum_{\mathbf{x} \in S(\mathbf{x})} (\mathbf{x}_c - \mathbf{x}) \tag{5}$$

is called Mean Shift Vector that can be expressed,using (4) as :

$$M_h(\mathbf{x}) = \frac{h^d}{d+2} \frac{\hat{\nabla} f(\mathbf{x})}{\hat{f}(\mathbf{x})} \tag{6}$$

The Mean Shift Vector at location $\mathbf{x}$ is aligned with the local density gradient estimate and is oriented towards the direction of maximum increase in density. For each point the Mean Shift Vector defines a path leading from the fixed point to a stationary point of estimated density where gradient is equal to zero.

4.2 Mean Shift Clustering for Personal Album

Given a generic point in the feature space(faces, backgrounds), the Mean Shift Vector in equation (5) describes a trajectory in the density space converging to points where the density is maximum. The set of all points converging to a local maximum is the *basin of attraction* for the found maximum density point. The procedure for the detection of modes in the data distribution is:

– Run mean shift to find stationary points for $\hat{f}(\mathbf{x})$
– Prune the found points retaining only the local maximum points

Clusters are refined through a merging procedure unifying adjacent clusters. Clusters are merged if:

$$\left\| \mathbf{y}_i - \mathbf{y}_j \right\| < \frac{h}{2} \tag{7}$$

where $\mathbf{y}_i$ and $\mathbf{y}_j$ are two local maximum points, $i \neq j$, and h is the bandwidth used to estimate the distribution density.

4.3 Entropy Based Clustering Measure

A number of evaluation indexes have been proposed to evaluate clustering methods, from the older Partition Coefficient and Partition Entropy [5] to the newest as partition based on exponential separation [24]. All of them tend to capture the goodness of the separation proposed by clustering. Typically these methods are oriented to fuzzy clustering more than to hard (crisp) clustering and they use a estimation of the density to evaluate the clustering performance (e.g. Parzen Windows). Since we already adopt a density estimation in the mean-shift procedure, to avoid a biased clustering measure, we choose to evaluate clustering from a set of hand annotated images that allow to evaluate the quality of label spreading in the set of samples. We define two indexes able to capture the clustering capability. The *Intra-Cluster Entropy* is defined as:

$$E_c = -\frac{1}{N_C * log(N_L)} \sum_{i=1}^{N_L} \sum_{j=1}^{N_C} \frac{u_{ij}}{T_j} log \frac{u_{ij}}{T_j} \tag{8}$$

where N_C is the number of clusters, N_L is the number of labels, u_{ij} is the number of times the $i-th$ label is present in the $j-th$ cluster and T_j is the number of labelled samples in the $j-th$ cluster. This function gives a measure of the entropy inside clusters. If many labels are present in a cluster the value u_{ij}/T_j is near the average and the *Intra-Cluster Entropy* is high. If a label is concentrated in few clusters and is absent in all the other the ratio u_{ij}/T_j is near 1 or near 0 and the entropy has a low value.

In figure 2 is shown the values of *Intra-Cluster Entropy* for the clustering with mean shift of a set of eigen-faces when the value of bandwidth is chosen among 4000 and 8000. For lower value of the bandwidth, the kernel covers a reduced volume and the number of modes is over-estimated. In this case the number of sample inside each cluster is reduced and the disorder is limited. With a higher bandwidth, the number of clusters decreases until all the samples are merged in a single cluster(see figure 4). In this case the Intra-Cluster Entropy reaches a maximum and will remain constant for higher values of the bandwidth. The second index, the *Intra-Label Entropy* is defined as:

$$E_l = -\frac{1}{N_L * log(N_C)} \sum_{i=1}^{N_L} \sum_{j=1}^{N_C} \frac{u_{ij}}{S_i} log \frac{u_{ij}}{S_i} \tag{9}$$

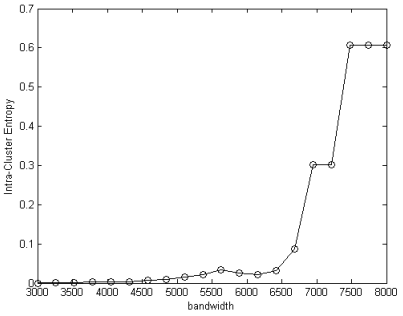


Fig. 2. Plot of *Intra-Cluster Entropy* for values of bandwidth from 4000 to 8000 for the clustering of eigenfaces

where N_C is the number of clusters, N_L is the number of labels, u_{ij} is the number of times the $i-th$ label is present in the $j-th$ cluster and S_i is the number of occurrence of the $i-th$ label. This function gives a measure of the distribution of a label across clusters. If a label is always present in a cluster, or in the opposite way always absent, the ratio u_{ij}/S_i is near 1, or near 0, and the entropy has a low value. On the other side if a label is generally present in many clusters, the more the value u_{ij}/S_i is near the average, the higher is the entropy.

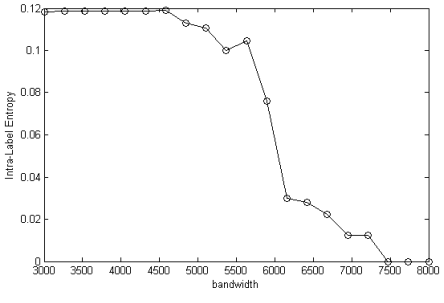


Fig. 3. Plot of *Intra-Label Entropy* for values of bandwidth from 4000 to 8000 for the Mean Shift clustering of eigenfaces

Similar consideration to the previous ones for *Intra-Label Entropy* can be drawn for this index.

Ideally, each label describing a set of samples, should be referred to a single cluster containing all uniform samples. In real cases this distribution is very rare due to the intrinsic variability of data and errors affecting sampling. Usually a tradeoff in the number of cluster must be fixed.

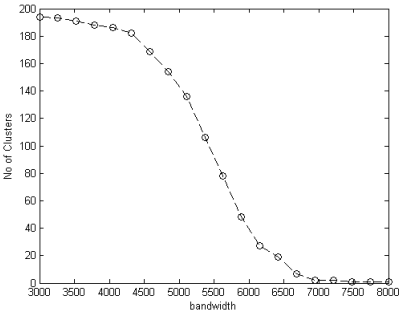


Fig. 4. Number of clusters determined with Mean Shift procedure on eigenfaces with values of bandwidth from 4000 to 8000

To modulate this tradeoff, a measure depending on *Intra-Cluster Entropy* and *Intra-Cluster Entropy* is defined and is called *Global Clustering Entropy*

$$E_G = \zeta \cdot \ E_c + (1 - \zeta) \cdot E_l \tag{10}$$

The value of the parameter ζ allows to modulate weight of *Intra-Cluster Entropy* and *Intra-Label Entropy* in the final clustering. The measure of *Global Clustering Entropy* referred to figure 2 and figure 3, with needed scaling, is shown in 5 considering ζ equal to 0.5.

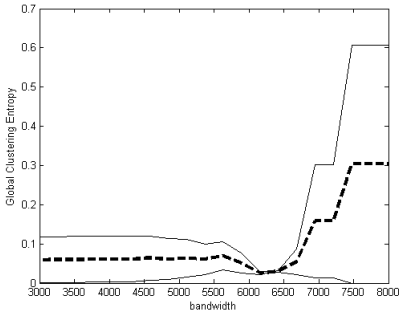


Fig. 5. Plot of the *Global Clustering Entropy*

Mean Shift Clustering for Composite Data. The clusterization of data through the mode seeking assumes the possibility to estimate distribution density with a single kernel being the data characterized by the same density distribution in all the space. In the case here considered, the sample in personal photo album can be split in multiple representation carrying orthogonal information composed together in a single data vector. Image data are represented with composite vectors merging multiple information that are processed with Mean Shift procedure. This technique has never been used to cluster data through

multiple information while in [8] a similar technique is used to segment images in homogeneous areas. Assuming that both domains (i.e. feature spaces) used to describe items of personal photo album allow the Euclidean norm as metric, a multivariate kernel is defined as product of two radially symmetric kernels:

$$K_{h_f,h_b}(\mathbf{x}) = \frac{C}{h_f^M h_b^P} k\left(\left\|\frac{\mathbf{x}^f}{h_f}\right\|^2\right) k\left(\left\|\frac{\mathbf{x}^b}{h_b}\right\|^2\right) \tag{11}$$

where $\mathbf{x}^f$ is the data in the first domain, $\mathbf{x}^b$ is the data referred to the second domain, h_f and h_b are the corresponding kernel bandwidths, C is the normalization constant. For personal photo album information is described, as shown in Section 3, as a composition of face representation and background representation and a multivariate kernel is applied to cluster information with Mean Shift procedure. The first part is referred to faces information and has a dimensionality f corresponding to the number of eigenfaces used. The second is background information with a dimensionality equal to b of the chosen features as shown in 3.2. The adopted kernels are two Epanechnikov kernel (2) each with a chosen value of bandwidth. Instead of evaluating empirically the performance of multiple values of the bandwidth, the *Global Clustering Entropy* introduced in the section 4.3 is used as performance measure. Driven by clustering results, the bandwidth value is automatically chosen. The process is run for both the domains, and ideally can be applied to all the set of orthogonal feature representing input samples, then the merging of clusters among multiple domains is performed (as in [8]):

- Run the Mean Shift procedure for the chosen domains fixing a value of bandwidth for each of them. The information about the convergence points are stored.
- Delineate in the joint domain the clusters by grouping the convergence points that are closer than the value of bandwidth in the corresponding domain. That is the basins of attraction of the corresponding convergence points are concatenated.
- Assign each point in the space to a cluster in the joint domain.

5 Experimental Results

To evaluate the performances of the proposed system we ran a set of experiments on a real photo collection. The digital album used is a subset of a real personal collection of 1008 images taken in the last three years. The presented process for face detection and rectification brought to the extraction of 331 images of rectified faces. The experiments have been aimed to the evaluation of the retrieval capability of the proposed system in terms of faces and background labeling but an entropy based analysis of the clustering process has also been performed to better understand the process itself.

We evaluated entropy as function of the bandwidth used in the clustering process and of parameters used in processing visual data, namely the α coefficient

of the sigmoid when processing the background data and the dimension of the eigenspace for face data. Since the variation of clusters composition according these parameters is smooth, the evaluation for a reduced set of combination of parameters produce significative results.

The background data analysis showed that the clustering providing the best value of Global Clustering Entropy was with a value of α equal to 0.95 and a bandwidth of 3.93. From the face data analysis we observed the best results for a dimension of eigenspace equal to 131 and a bandwidth of 5756. We also evaluated as the dimension of the eigenspace affect the number of clusters (see Fig. 8) and results where coherent with hypothesis made with our model.

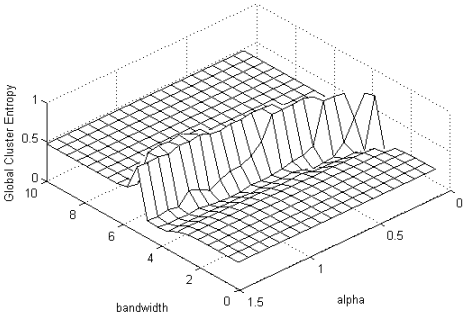


Fig. 6. Plot of the *Global Entropy* for the clustering of background data as function of the bandwidth and the α coefficient of the sigmoid

To evaluate our approach with respect to the semantic meaning of clusters we divided all the images of the test collection in six categories (*beach*, *indoor*, *nature*, *public garden*, *snow*, *urban*) representing six typical contexts mainly present

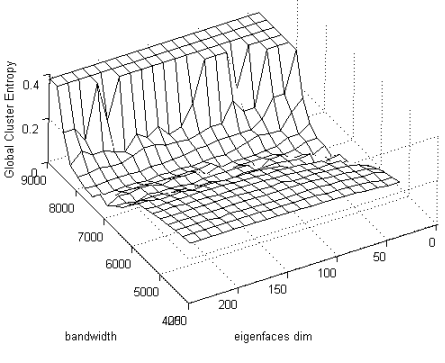


Fig. 7. Plot of the *Global Entropy* for the clustering of faces data as function of the bandwidth and the dimension of the eigenspace

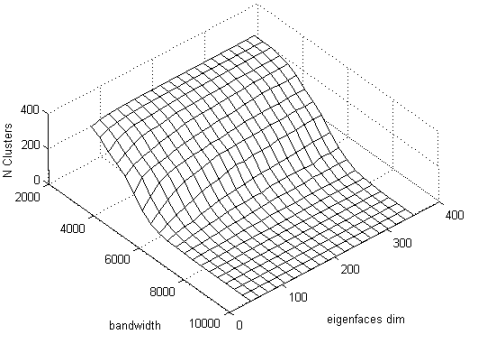


Fig. 8. Plot of the number of clusters as function of the bandwidth and dimension of the eigenspace

in the collection and assigned an identifier to the four people present in most of the photos.For the first experiments, data are clustered using a single domain information. For each domain the optimal point according to the Global Clustering Entropy (eq. 10) is chosen. The results for the clustering of background are shown in the table 1.

Table 1. Percentage occurrence of labels in generated clusters

	beach	indoor	nature	public garden	snow	urban
Cl 1	11%	32%	2%	40%		15%
Cl 2		89%		11%		
Cl 3		96%		4%		
Cl 4		100%				
Cl 5		13%		63%		25%
Cl 6	6%	42%		52%		
Cl 7		100%				
Cl 8		100%				
Cl 9				67%		33%

For faces data, the optimal point according the entropy measure is found for a bandwidth of 5756 and a dimension of the eigenspace equal to 131. Discarding all the cluster with a single element the number of remaining clusters is equal to 6 and the distribution is shown in table 2. The id from 1 to 4 are the most recurrent in image repository.

The clusters for the personal album using information from both domains are created using the procedure described in 4.3. An evaluation of this clusterization is achieved calculating the Global Clusterization Entropy (eq. 10) using labels given by couples (*identity*, *context label*). In the table 3 values of Entropy are shown when are varied values of clusterization bandwidths.

Table 2. Percentage occurrence of identities in generated clusters

	Id 1	Id 2	Id 3	Id 4	Other
Cl 1		77%	9%	5%	9%
Cl 2	3%	22%	53%	6%	17%
Cl 3	22%	33%	33%		11%
Cl 4	11%	11%	44%		33%
Cl 5			100%		
Cl 6			100%		

Table 3. Value of global entropy for clusterization

	3.0	3.5	4.0	4.5	5.0
4756	2.67	2.70	2.78	2.87	3.10
5256	2.70	2.72	2.82	2.92	3.11
5756	3.03	3.10	3.20	3.22	3.19
6256	4.31	4.87	5.89	5.71	5.45
6756	6.41	8.31	10.09	9.78	8.07

Along the columns of the table the value of the bandwidth for clustering of faces varies from 4756 to 6756 with even intervals. Along the columns the bandwidth varies from 3 to 5 with even intervals.

6 Conclusions

A novel approach to cluster composite data driven by a clusterization measure has been presented. The approach has been demonstrated on the very interesting problem of automatic organization of photos in personal album. To demonstrate the approach we developed techniques to represent images as composite data corresponding to foreground and background. In our experiments data in collection of photos are represented in two spaces retaining information about people in the picture and low level features related to the context of the picture itself. Due to the domain constraint we were able to use computer vision technique to extract information automatically and with reasonable correctness. Results of experiments on a real set of a thousand pictures based both on entropy measures of the cluster and on comparison to manually provided ground truth are very promising.

Future work include exploiting more image information automatically computed from image data. Other information as clothing characterization of people captured, more detailed information about backgrounds (e.g. multiple feature and color descriptors), time of capture and other hints from EXIF data could be easily added in the proposed framework and we expect even better results.

References

1. Caminati, L., Don, A., et al.: Detection of visual dialog scenes in video content based on structural and semantic features. In: Proc. of International Workshop on Content-based Multimedia Indexing, CBMI (2005)
2. Abdel-Mottaleb, M., Chen, L.: Content-based photo album management using faces' arrangement. In: IEEE International Conference on Multimedia and Expo., ICME (2004)
3. Ardizzone, E., La Cascia, M., Vella, F.: A novel approach to personal photo album representation and management. In: SPIE, vol. 6820 (2008)
4. Berg, T.L., Berg, A.C., Edwards, J., Maire, M., White, R., Teh, Y.W., Learned-Miller, E., Forsyth, D.A.: Names and faces in the news. In: Proc. of IEEE International Conference on Computer Vision and Pattern Recognition, CVPR (1994)
5. Bezdek, J.C.: Pattern Recognition with Fuzzy Object Function. Plenum (1981)
6. Chen, J.Y., Bouman, C.A., Dalton, J.C.: Hierarchical browsing and search of large image databases. IEEE Transaction on Image Processing 9(3), 442–455 (2000)
7. Cheng, Y.: Mean shift, mode seeking and clustering. IEEE Transaction on Pattern Analysis and Machine Intelligence, 790–799 (August 1995)
8. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Transaction on Pattern Analysis and Machine Intelligence, 603–619 (May 2002)
9. Cui, J., Wen, F., Xiao, R., Tian, Y., Tang, X.: Easyalbum: An interactive photo annotation system based on face clustering and re-ranking. In: Proc. of ACM CHI (2007)
10. Cuiy, J., Wenz, F., Xiaoz, R., Tianx, Y., Tang, X.: Easyalbum: An interactive photo annotation system based on face clustering and re-ranking. In: Proc. of CHI (2007)
11. Deng, D.: Content based comparison of image collection via distance measuring of self organized maps. In: Proceedings of 10th INternational Multimedia Modelling Conference (2004)
12. Girgensohn, A., Adcock, J., Wilcox, L.: Leveraging face recognition technology to find and organize photos. In: Proc. of ACM MIR (2004)
13. Goldberg, J., Gordon, S., Greenspan, H.: Unsupervised image-set clustering using an information theoretic framework. IEEE Transaction on Image Processing (2), 449–458 (2006)
14. Graham, A., Garcia-Molina, H., Paepcke, A., Winograd, T.: Time as essence for photo browsing through personal digital libraries. In: Proc. of ACM JCDL (2002)
15. Kang, H., Shneiderman, B.: Visualization methods for personal photo collections: Browsing and searching in the photofinder. In: Proc. of IEEE International Conference on Multimedia and Expo., ICME (2000)
16. Krishnamachari, S., Abdel-Mottaleb, M.: Hierarchical clustering algorithm for fast image retrieval
17. Lee, B.N., Chen, W.-Y., Chang, E.Y.: A scalable service for photo annotation, sharing and search. In: Proc. of ACM International Conference on Multimedia (2006)
18. Li, C.-H., Chiu, C.-Y., Huang, C.-R., Chen, C.-S., Chien, L.-F.: Image content clustering and summarization for photo collections. In: Proceedings of ICME, pp. 1033–1036 (2006)
19. Naaman, M., Yeh, R.B., Garcia-Molina, H., Paepcke, A.: Leveraging context to resolve identity in photo albums. In: Proc. of ACM JCDL (2005)

20. Shyu, M.-L., Chen, S.-H., Chen, M., Zhang, C.: A unified framework for image database clustering and content-based retrieval. In: ACM International Workshop On Multimedia Databases archive Proceedings of the 2nd ACM international workshop on Multimedia databases, pp. 19–27 (2004)
21. Song, Y., Leung, T.: Context-aided human recognition clustering. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3953, pp. 382–395. Springer, Heidelberg (2006)
22. Spyrou, E., Kapsalas, P., Tolias, G., Mylonas, P., Avrithis, Y., et al.: The cost292 experimental framework for trecvid 2007. In: Proc. of 5th TRECVID Workshop (2007)
23. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proc. of IEEE International Conference on Computer Vision and Pattern Recognition, CVPR (2001)
24. Wu, K.L., Yang, M.S.: A cluster validity index for fuzzy clustering. Pattern Recognition Letters, 1275–1291 (2005)
25. Zhang, L., Chen, L., Li, M., Zhang, H.: Automated annotation of human faces in family albums. In: Proc. of ACM International Conference on Multimedia (2003)
26. Zhang, L., Hu, Y., Li, M., Ma, W., Zhang, H.: Efficient propagation for face annotation in family albums. In: Proc. of ACM International Conference on Multimedia (2004)

Towards a Fully MPEG-21 Compliant Adaptation Engine: Complementary Description Tools and Architectural Models*

Fernando López[1], José M. Martínez[1], and Narciso García[2]

Grupo de Tratamiento de Imágenes
[1] Escuela Politécnica Superior. Universidad Autónoma de Madrid. E-28049. Spain
{f.lopez,josem.martinez}@uam.es
[2] E. T. S. I. Telecomunicación. Universidad Politécnica de Madrid. E-28040. Madrid. Spain
narciso@gti.ssr.upm.es

Abstract. This paper develops a number of aspects of metadata-driven adaptation within the MPEG-21 framework that we consider are not clearly covered in the standard. Specifically, the paper introduces three complementary description tools that we have made use of and that we consider would fill certain gaps in the standard. Besides, this paper offers a group of architectural design patterns along with a pragmatic group of considerations that we have gathered during the development of our MPEG-21 compliant adaptation engine.

Keywords: multimedia, adaptation, mpeg-21, description tools.

1 Introduction

MPEG-21 [1] is a multimedia framework proposed by the ISO (International Organization for Standardization) to specify a complete set of *description tools* that pave the ground for end-to-end solutions in multimedia systems. Within this framework, multimedia elements are represented using DIs (Digital Items) [2]. According to this framework a DI can convey *resources* (media files) and *descriptors* (metadata). The DI is a generic container capable of representing a great variety of multimedia. MPEG-21 Part 7 [3] standardizes a group of *description tools* capable of collecting the information that an adaptation engine can draw on to drive the adaptation of those DIs.

During the last years we have been working on the development of an adaptation engine named CAIN (Content Adaptation INtegrator) [4]. Numerous aspects have been elucidated since the first implementation. The implementation that we discuss in this document complies more properly and brings into play a wider range of description tools proposed by the MPEG-21 framework. Throughout its implementation we

* Work supported by the European Commission (IST-IP-001765 – aceMedia, IST-FP6-027685 – MESH), Spanish Government (TEC2007-65400 - SemanticVideo) and Comunidad de Madrid (S-0505/TIC-0223 - ProMultiDis-CM). The Ministerio de Educación y Ciencia of the Spanish Government by means of the FPU fellowship issued to the first author has also supported this work.

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 155–169, 2010.

have encountered a number of important difficulties. We consider helpful to highlight these issues for people involved in the construction of multimedia adaptation systems, especially if they are determined to provide MPEG-21 interfaces to their users. The clarification of these difficulties may also be useful for people who intend further interaction with other MPEG-21 compliant multimedia systems.

Section 2 revises the state of the art regarding multimedia automatic adaptation taxonomies and techniques. Section 3 introduces CAIN. Section 4 offers three new description tools that we consider would fill a few gaps that we have identified in the standard. Section 5 describes the architectural models that we have employed during the development of CAIN. Following the architecture of CAIN, section 6 motivates the usefulness of describing its adaptation capabilities with another group of description tools. Finally, section 7 provides the conclusions obtained from the development of CAIN. We expect that the ideas gathered in this document would be helpful for future advances in multimedia systems within the MPEG-21 framework.

2 State of the Art

Nowadays a great deal of effort is being put into developing effective techniques to automatically adapt multimedia content to the usage environment. Roughly we can divide the automatic adaptation problem into two parts: first, the decision of which adaptation to perform and second, the execution of the selected adaptation. However the proposed techniques in the literature widely vary in nature, features under consideration and modus operandi. Certain techniques take into account a description of the environment [5][6], and model the adaptation decision problem making use of techniques like constraints satisfaction [5] or automatic planning [6]. Other techniques are concentrated on analysing the resource to measure the utility or the quality of the adaptation [7]. Usually there exists a distinction between mandatory constraints and user's preferences [8].

Most of the authors have made use of MPEG-21 Part 7 UED (Usage Environment Description) [3] to represent the features of the usage environment. Due to the all-purpose nature of MPEG-21 descriptions, we have selected this technology to investigate the multimedia decision problem. However, since we are only choosing a set of features, such selected features described by means of the MPEG-21 description tools in this work –could be described without difficulty using another description technology. Thus, the ideas proposed in this document could be easily employed in adaptation engines that are focused on different base description technologies.

2.1 Multimedia Composition Levels

In computer science, *media* is a piece of information prepared to be consumed by the human senses. Regular media is represented only in one *modality* (such as video, audio, images or text). *Multimedia* extends the notion of media with the combination of at least two media elements synchronized for presentation. Multimedia may be seen at different (no necessarily orthogonal) levels of composition:

1. *Resource level*. Isolated and homogeneous multimedia content. It can be consumed alone or in conjunction with other multimedia content. Usually (but not always) it is stored in a file. We can further divide multimedia at this level into: a) single *media level*, where a standardization body defines the whole format of a single media modality (e.g. .jpg or .mp3), and b) *multimedia level*, in which the resource is made up of one or more media modalities (e.g. .mpg or .html)
2. *System level*. Different kinds of resources (such as media, multimedia or even entire scenes) are gathered together forming high-level structures and concepts (e.g. MPEG-21 DIs [2] or NewsML [9]). Usually this level also comprises metadata about the resources (e.g. MPEG-7 *MediaInformation* [10] or semantic information).
3. *Scene level*. Occurs when the multimedia elements (single or composed media elements) form a whole multimedia presentation. This level extends the system level with: a) *layout information*, which provides cues, restricts or completely defines the way in which the content must be rendered onto the screen of the terminal (e.g. HTML), and b) *synchronization information*, which helps during the rendering of the content along the timeline as well as to synchronize the different elements of the scene (e.g. SMIL, MPEG-4 BIFS or Adobe Flash).

In this paper we present results with respect to system level adaptation of MPEG-21 DIs. Scene level adaptation (spatial and temporal adaptation) along with the rendering of the DIs in the scene are out of the scope of this work. You can consult [12] for information about scene level adaptation of DIs.

2.2 DIA Description Tools

This section surveys two *DIA (Digital Item Adaptation) Description Tools* proposed in MPEG-21 Part 7. Section 0 will introduce an alternative *DIA Description Tool* that we propose to be useful in order to drive the adaptation, and we will illustrate a suitable use case where this new *DIA Description* is thoroughly applicable.

Usage Environment Description Tool

The MPEG-21 Part-7 [3] standard has defined, among others, the *Usage Environment Description* (UED) *Tools*, which is a set of description tools widely used to describe the target environment of the adaptation. Instances of the *UED Tools* are referred as *UED Descriptions*, and with regard to the UED there exist four main *UED Descriptions*: *Terminals*, *Networks*, *Users* and *NaturalEnvironments*. Since the standard allows each *UED Description* to be instantiated from zero to *n* times, the UED acts as a database where the features of different usage environments are described.

DIA Configurarion Tool

The *DIA Configuration* (DIAC) *Tool* [3] is a tool that allows guiding the adaptation considering the DI author's intentions. This information is conveyed at DI level in a *Descriptor* element. Specifically two methods to provide DIAC information are

available. The first method is through the use of the *UserSelection /BackgroundConfiguration* elements, which allows the DI's author to suggest the means by which *Choice*/*Selection* of the DI should be processed. The second method is through the use of the *SuggestedDIADescriptions* used by the author of the DI to provide XPath [13] expressions pointing to the fragments of the *DIA Description* to be used during the adaptation.

The *DIAC Tool* proposes the existence of a negotiation protocol between the DI server and DI client. When the DI client requests a DI to the DI server, the DI server sends a DI alo4ng with a *DIAC Description* to the DI client. The *DIAC Description* can be used to decide where to perform the adaptation: client, proxy and server. A combination of them is also possible.

3 Introduction to CAIN

CAIN [4], our MPEG-21 adaptation engine implementation, is designed to perform all the levels of adaptations described in section 0. Since CAIN is a metadata-driven adaptation engine, it makes use of annotations on the resources to drive the adaptations. Specifically, multimedia content is annotated using the MPEG-7 [10] standard, and the adaptation system makes use of the MPEG-21 framework description [1]. The inputs and outputs of CAIN are represented as MPEG-21 DIs (Digital Items) [2]. The main reasons that justified our selection of MPEG-21 DIs is that (1) they are capable of representing virtually every kind of media or system level composition and (2) that they allow for the annotation of the multimedia resources. In particular: media level adaptation can be performed referencing a media resource in the *Resource* element of its *Component* element. Moreover, this *Component* may include a *Descriptor* element with metadata about the resource. In the case of structure level adaptation the *Component* elements of the DI can be modified as needed.

Fig. 1 shows an example of a DI that includes several *Component* elements. In this figure, there exists a *Component* with a HTML web page, which has references to video and image resources in other *Component* elements. According to the taxonomy given in section 0, the image is a media level resource, the video is a synchronization level resource, and the HTML page is a system level structure composition that has references to the video and image resources. In this scheme, when the DI is adapted, both the web page and the linked resources are adapted. In the case of layout adaptation an HTML or SMIL file is conveyed in the *Descriptor* of the *Component* of the DI. The usage environment restrictions (such as screen size) are used to perform the layout level adaptation. Synchronized level adaptation can be applied to transcoding or transforming multimedia level synchronized resources (e.g. .mpg files).

As CAIN is designed to be capable of adapting every kind of media and multimedia document, hereafter the term *resource* will refer to the file referenced by the *Resource* element. This resource may be a media resource (e.g. .jpg) or a multimedia resource (e.g. .mpg or .html).

We will use the term *DI adaptation* to refer to the type of adaptation performed by CAIN over a DI. The input to a DI level adaptation is a DI, and the output of the adaptation process is represented as an *adapted DI*. As a DI could comprise one or more

Component elements, we have decided that adaptation within CAIN is going to be performed at *Component* level by software tools named *Component Adaptation Tools* (CATs). We also have considered dependencies in situations like the one plotted in Fig. 1. In this situation the resource of a *Component* (the HTML page) includes references to the resources of other *Component*s.

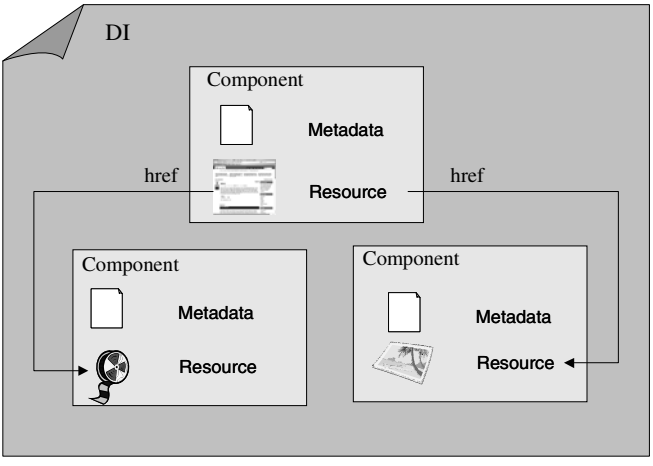


Fig. 1. DI composition within the CAIN

Metadata may be automatically extracted by the CATs, or may be provided as input by the *Components*. Usually *Component* level adaptation is performed in two steps: During the first step the resource is retrieved from the media repository. In addition, if semantic adaptation is used, the resource and associated metadata may be analysed. During the second step metadata conveyed in the *Component* and the information that results from analysing the content is used to adapt (transcode, transform or transmode) the resource and associated metadata.

4 Describing the Configuration of the Adaptation Engine

This section takes on the rationalization of the solution that we suggest for reusing the MPEG-21 description of the content and context onto different multimedia adaptation requests. The upshot of this discussion is in section 0 where we will offer the use of an MPEG-21 complementary DIA *Description Tool*.

4.1 Content DI, Context DI and Configuration DI

Authors (such as [11]) have modelled their adaptation engines assuming the existence of two kinds of DIs: the Content DI and the Context DI. The *Content DI* is a DI that carries out the multimedia document as well as the DIAC. The *Context DI* contains a description of the UED.

We consider very useful the exploitation of the idea behind the *DIAC Tool* whenever an adaptation is going to be performed. This is because (as indicated in section 0) the UED allows the description of multiple usage environments, and hence it is important to know which environment is going to be the target of the adaptation. Although there exist scenarios where the DIAC mechanism is applicable, we have identified two drawbacks in the DIAC mechanism: (1) there exists a coupling between the Content DI (with the multimedia document and DIAC) and the Context DI (with the UED). That is, the Content DI encloses the alternative adaptation options to the usage environment, and as a consequence, the Content DI document must be modified when the usage environment capabilities (Context DI) change. (2) The DIAC mechanism assumes that the possible usage environments are known when the Content DI is created.

We propose that in order to avoid the first drawback it is useful to make use of three DIs: the *Content DI* (with the multimedia document), the *Context DI* (that acts as a database where the different terminals, networks, users and natural environments under consideration are described), and the *Configuration DI* that encloses the DIAC. The exact way to represent this configuration is the innovative part of this section. In the next section we are going to rationale it. The main aim of our proposal is that the Content DI will not be modified when the usage environment (located in the Context DI) changes. The Configuration DI also solves the second drawback: the Content DI and Context DI could be created and stored into the MPEG-21 multimedia system during the adaptation engine implementation. The Configuration DI would be created only when the adaptation engine has been deployed and is ready to start serving adaptation requests.

4.2 The Adaptation Request Configuration Description Tool

Currently the MPEG-21 Part 7 standard proposes two different *DIAC Descriptions*: (1) The *UserSelection*/*BackgroundConfiguration* elements that indicate whether the DI *Choice*/*Selection* mechanism must be presented to the user or automatically decided by the system, and (2) the *SuggestedDIADescriptions* where the DI's author suggests which *DIA Descriptions* should be used to make decisions. As indicated in section 0, both *DIAC Descriptions* assume the existence of a negotiation mechanism and place the *DIAC Descriptions* in the DI that is going to be consumed. We propose that under certain circumstances (for example when the adaptation engine is implemented as a middleware that provide an API instead of as a network agent) it is useful to relax the previous assumptions and make use of a third (no considered in MPEG-21) *DIAC Description Tool* that we will refer as *Adaptation Request Configuration* (ARC) *Description Tool*. The *ARC Description Tool* is in charge of selecting a zero or one instance value of the *Terminals*, *Networks*, *Users* and *NaturalEnvironments* available in the UED. For example, one adaptation request could be intended to adapt one Content DI to a mobile *Terminal* and another adaptation request could be intended to adapt the same Content DI to a laptop *Terminal*. Note that the Content DI

and Context DI could be created before deploying the adaptation engine in a multimedia system. However the set of feasible *ARC Descriptions* depends on the *UED Description* of the multimedia system where the adaptation engine has been deployed, and therefore the *ARC Description* will not be created until the adaptation engine is deployed in a particular multimedia system.

4.3 DIs and UED in CAIN

In this section we introduce the way in which the Content DI and Context DI are implemented in CAIN. When CAIN is deployed in a multimedia system, we presuppose the existence of multimedia represented as a group of Content DIs. In section 0 we will explain that CAIN is also capable of representing non-MPEG-21 multimedia content as DIs before performing the adaptation. For example, *video1.xml*[1] is a Content DI with only one *Component* element that includes a *Resource* element that points out to a video resource and a *Descriptor* element where the resource is annotated in MPEG-7.

An example of a Context DI can be found in the *ued.xml* file. The document contains the *Terminals*, *Networks* and *Users* elements, which have an associated unique ID. These IDs will be used in the next section to indicate which *Terminal*, *Network* and *User* to must be used during the adaptation.

4.4 ARC Driven Adaptation Example in CAIN

Before a DI level adaptation could be executed in CAIN, an *ARC Description* must be provided. If the *ARC Description* is not specified, CAIN is not capable of adapting the Content DI, because there is no way to decide the transformations to perform over the elements of the Content DI. Listing 1 shows an example of an *ARC Description*. Within the ARC, the target *Terminal* description contains basic information (such as *CodecCapabilities*, *Display*, ...) essential to decide the adaptation that should be performed, so we have marked the *Terminal* element as mandatory (see *catc.xsd*). In Listing 1 the *Terminal* with ID *mobile_1* has been selected. The description of this *Terminal* is located in the *ued.xml* document. The two other *UED Descriptions* that CAIN supports (*Network* and *User*) have been left optional (see *catc.xsd*), although they are useful to reach a better adaptation. The *Network* and *User* elements of Listing 1 are optional and, if present, they would contain respectively the ID of the *Network* and *User* of the Context DI to use during the adaptation request. CAIN does not make, currently, use of the *NaturalEnvironment* descriptions.

4.5 Operation Modes

In addition to the ID of the *UED Descriptions* to use during the adaptation, the *ARC Description* is able to describe what we refer to as *operation modes*. An *operation*

[1] Due to space constraints, at http://www-vpu.ii.uam.es/publications/dt_arch/ we provide a number of documents online.

mode allows selecting parts of the *UED Description* whenever several modes could be used. For instance, a *Terminal* might support different screen resolution modes. In this case the adaptation engine must be capable of selecting one of these operation modes, but additionally the *ARC Description* can be used to signal a specific operation mode that the adaptation engine must obey.

```
<?xml version="1.0" ?>

<DIA xmlns="urn:mpeg:mpeg21:2003:01-DIA-NS"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<Description xsi:type="ARCType">
  <Terminal>
   <Id>mobile_1</Id>
   <OperationMode>
   //TerminalCapability[@type='dia:DisplaysType']/Display/
   DisplayCapability[@type='dia:DisplayCapabilityType']/Mode/
   Resolution/[@id="176x144"]
   </OperationMode>
  </Terminal>
  <Network>
   <Id>ethernet</Id>
  </Network>
  <User>
   <Id>reporter</Id>
  </User>
 </Description>
</DIA>
```

Listing 1: *ARC Description* example

5 Modular Architecture of CAIN

This section provides the architectural model of CAIN. The purpose of this section is twofold: (1) to suggest a group of software design patterns that may result useful to future adaptation engine implementors, and (2) to explain the applicability of the notion of CATs (introduced in section 0), which are reusable software tools that could be put into operation in different adaptation scenarios. After these concepts will be clarified, section 0 presents two description tools that we consider helpful to make adaptation decisions.

5.1 CAIN Modules

CAIN serves requests through two external programming interfaces: (1) *The media level transcoding interface* is concentrated on performing blind adaptation of a media resource. In addition to media level, this interface is also capable of performing system level adaptation of videos composed of one or more audio and visual streams. (2) *The DI level interface* is concentrated on performing system level (semantic or blind) adaptations where metadata is used for helping during adaptation.

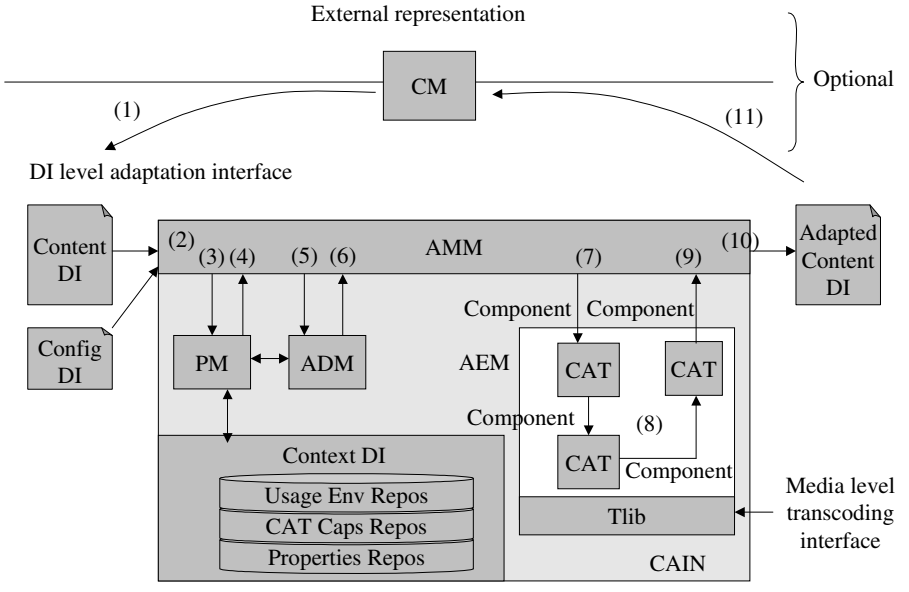


Fig. 2. Control flow within CAIN

The CAIN engine has been functionally divided into modules. These modules and the control flow along the adaptation process are depicted in Fig. 2. The media level transforming interface is served through a conventional software library that we have implemented named *Tlib*. This library implements blind image transcoding and transforming, video transcoding and transforming, video truncating and video frame extraction. The CATs are software modules that perform (semantic or blind) *Component* level adaptation. As each *Component* element includes both the resource and its metadata, the CATs provide parts of the DI level interface. The CATs take advantage of the *Tlib* to perform parts of the blind adaptation of the resources referenced in the *Components*. In addition, the CATs adapt the metadata of the *Component* elements. Note that whereas CATs are capable of performing (semantic or blind) *Component* transcoding, transforming, transmoding and summarization adaptations, the *Tlib* is concentrated on achieving blind media level operations (e.g. transcoding MPEG-2 video with MP@ML stream video and MPEG-2 audio stream to MP4 video container with AAC/H264 streams).

The Context DI encompasses a number of repositories. The *Usage Env Repository* is a set of documents annotating the usage environments using the *UED Tools* (introduced in section 0). The *CAT Capabilities Repository* is another set of documents annotating the *conversions* that the available CATs are capable of performing. Each conversion has a set of valid input and output properties along with its corresponding values. Every CAT must have an associated *CAT Capabilities* document where the different conversions that the CAT is capable of accomplishing are described. The *Properties Repository* is yet another set of documents committed to relate the CAT Capabilities and the UED. These relationships can be represented as properties that

take one or more values. Specifically, we use XPath expressions to represent the values of those properties. In addition, it's important to remark that there must exist one CAT Capabilities file per CAT, but the Properties Repository stores properties that are common for the CATs altogether, and in this sense, all the CAT implementers must provide these properties (e.g. *media_system_format*, *visual_coding* or *visual_height*). Some of these properties are marked as mandatory and the rest as optional.

The *Parsing Module* (PM) is the module in charge of loading and querying the Content DI, Configuration DI and Context DI. Within the PM we also find the *Coupling Module* (CM). There exist a wide variety of multimedia representation standards and systems. CAIN is a software engine that is designed with the ability to be integrated in heterogeneous environments. The CM is the module interfacing with other components that may be using *external technology* (i.e., non-MPEG-21 technology) to represent multimedia (e.g. HTML, SMIL or NewsML). The main purpose of the CM is to enable the integration of CAIN adaptation services into those heterogeneous multimedia systems. The CM is in charge of transforming that external representation of multimedia onto an MPEG-21 compliant input DI that CAIN is capable of processing. In addition, the CM is in charge of transforming the adapted output DI onto its external representation.

When a DI arrives, the *Adaptation Decision Module* (ADM) is in charge of deciding the *sequence of conversions* committed to transforming the initial Content DI onto an adapted Content DI taking into account the information in the Configuration DI and Context DI. Different CATs sequentially perform the conversions of the sequence. A CAT receives as input a Content DI and generates as output another Content DI so that a conversion operation has been performed over one or more of its *Components*. The module in charge of coordinating the CATs execution is the Adaptation Execution Module (AEM).

The *Adaptation Management Module* (AMM) is in charge of coordinating the whole Content DI level adaptation process. Modules below the AMM are concentrated on performing different tasks requested by the AMM. It's important to remark that, although the input/output unit of the DI level interface is the Content DI, the adaptations are always performed over one or more *Components* of the Content DI, and not over the entire Content DI. In this sense the DI acts as a container of *Components*. Before adaptation could be performed, CATs must be installed in the AEM and its conversion capabilities registered in a *CAT Capabilities Repository*. Besides, information about terminal capabilities, network descriptions, user characteristics and preferences must be registered in the *Usage Environment Repository*.

5.2 Control Flow

Numbers in Fig. 2 provide the temporal order (control flow) of the tasks involved in the adaptation process. (1) The CM is the module in charge of dealing with the specific environment where CAIN has been deployed in order to provide its multimedia adaptation services. Different instances of the CM are interchangeable modules created to interact with the specific multimedia representation details of the deployment system. They transform the external multimedia representation of the environment onto MPEG-21 compliant DIs that CAIN is capable of processing. (2) When a

Content DI along with a Configuration DI arrives (via the DI level adaptation interface), they are sent (3) to the PM to represent that information in the computer memory as objects. The PM is in charge of reading the *DIA Descriptions* (ARC, UED, CAT Capabilities and Properties). The information in those documents is cached to improve performance. (4) This information is returned to the AMM, and (5) the information is sent to the ADM that decides which conversions (represented as a sequence of conversions with the corresponding set of proper parameters) will be used to achieve the best adaptation. Specifically, the sequence of conversions returned by the ADM is the sequence of conversions that should be executed over the *Component* elements of the Content DI, and not over the entire Content DI. (6) The selected sequence of conversions is sent back to the AMM. If the decision returned by the ADM is that the adaptation cannot be performed (given the input Content DI, Configuration DI, available CATs and usage environment), the AMM must inform of that condition, and the adaptation process finishes. Assuming that the ADM returns a valid sequence of conversions, (7) this sequence is transmitted to the AEM, which executes the consecutive sequence of corresponding CATs. (8) During the CAT's execution, optionally, CATs may append information to the *Descriptor* element of the *Component* so that subsequently CATs could use it. We use the name *static decisions* to refer to the usage environment and user preferences based decisions, because they don't depend on the resource content (only the resource description), and the ADM can make those decisions. On the contrary, we use the term *dynamic decisions* to refer to adaptation decisions that perform measures over the resource content, and cannot be taken until the resource is available (such as utility and quality based decisions [7]). As explained in [14], CATs are non-deterministic and they are intended to make the dynamic decisions. (9) When all the conversions of the sequence have been executed, the AEM returns the adapted DI to the AMM. (10) CAIN uses the DI level adaptation interface to return the adapted Content DI to the caller. (11) Eventually, the adapted Content DI may need to be transformed to an external representation. In this case the CM receives the adapted Content DI and transforms it into an external representation.

6 Description of the Adaptation Capabilities

In [15] we introduced the notion of CATs (see section 0) and its associated CAT Capabilities (see section 0). In this section we are going to provide a number of enhancements over the CAT Capabilities that we have identified and implemented into CAIN. In particular, there exist two main changes with respect to the CAT Capabilities model previously proposed in [15]: (1) Each CAT can incorporate several *conversions*[2]. This is due to the fact that the conversion capabilities that can be performed are not always easy to describe if we do not break apart the capabilities of an individual CAT into several conversion capabilities elements. For example, if a specific CAT is capable of accepting JPEG and PNG images, but PNG images are accepted only in

[2] This term has been selected in accordance with the term *Conversion Tool* that MPEG-21 Part 7 Amendment 1 proposes to represent the software or hardware element that performs the adaptation.

greyscale, whereas JPEG images are accepted in colour and greyscale. In this case the CAT capabilities must be split into two separate conversion capabilities: one conversion capabilities stating that PNG images are accepted in greyscale and another conversion capabilities stating that JPEG images are accepted in both colour and greyscale. (2) As explained in [14], within CAIN the *potential adaptation capabilities* are represented as multivaluated properties, i.e., properties that can take several possible values (e.g. *format*={*mpeg*-1, *mpeg*-2, *mpeg*-4}). We have modified the description of the conversions so that each input and output property can take multiple values.

```
<?xml version="1.0" ?>
<dia:DIA xmlns="urn:gti:cain-cat-capabilities"
         xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-NS"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <dia:Description xsi:type="CATCapabilitiesType"
                           id="video_transcoder_cat">
  <!-- Offline conversion using the ffmpeg command -->
  <ConversionCapability xsi:type="ConversionCapabilityType"
                                      id="mpeg4_offline_transcoder">
   <Preconditions>
    <URL>
     <AnyValue/>
    </URL>
    <MIMEType>
     <ValueSet>
      <Value href="video/mpeg">MPEG video</Value>
      <Value href="video/mpg">MPEG video</Value>
      <Value href="video/avi">AVI video</Value>
     </ValueSet>
    </MIMEType>
    <Bitrate>
     <RangeValueSet from="40000" to="2000000"/>
    </Bitrate>
   ..............................
   </Preconditions>
   <Postconditions>
    <URL>
     <AnyValue/>
    </URL>
    ..............................
   </Postconditions>
  <!-- Online conversion using the ffmpeg libraries and JNI -->
  <ConversionCapability xsi:type="ConversionCapabilityType"
                                      id="mpeg4_online_transcoder">
  ..............................
  </ConversionCapability>
 </dia:Description>
</dia:DIA>
```

Listing 2: *CAT Capabilities Description* example

The CAT Capabilities Tool that we propose is available online in *ccatc.xsd*. The *CATCapabilitiesType* type represents a particular CAT, and the *ConversionCapabilityType* type represents each conversion that the CAT is capable of performing. It's

worth to notice that the *CATCapabilitiesType* is defined as a derivation by restriction of the MPEG-21 *DIADescriptionType*, and hence the *CATCapabilitiesType* can be seen as a non-MPEG-21 standardized DIA *Description Tool*. For instance, Listing 2 shows the more relevant parts of a video transcoding CAT example available online in *video_transcoding_cat.xml*.

This CAT comprises two *ConversionCapability* elements named *mpeg4_offline_transcoder* and *mpeg_online_transcoder*. The *Preconditions* and *Post-conditions* elements convey information related to the media formats that the conversion respectively accepts and produces. Those inputs and outputs correspond with the preconditions and postconditions of the conversion according to the model that we have proposed in [14].

6.1 The Properties Repository Description Tool

In early versions of CAIN, the PM was in charge of parsing the different DIs. For each document, it produced a hierarchical tree of objects. After that the ADM implementation went through those hierarchies searching for the specific values needed to make decisions about the adaptations to perform. This course of actions proved tedious to implement and difficult to maintain since changing one element of an individual document implied searching for the use of this property in the algorithm of the ADM and updating the algorithm in one or more points.

Those difficulties motivated the idea of gathering the relevant properties to be used by the ADM in the Properties Repository. Each property is represented as a label with an associated XPath expression that points out to the specific part of the DIs or CAT Capabilities where their values are located. Even though the PM is still in charge of parsing the documents and loading them in memory as hierarchies of objects, the access to the properties is no longer performed directly by the ADM. Instead, those values are accessed by means of properties stored into the Properties Repository. In this way, modifications in the location or in the number of the properties are done by changing the Properties Repository, instead of changing the ADM source code.

The Properties Repository scheme that we propose is available online in *cpr.xsd*. The *PropertiesReposType* type is also defined as a derivation by restriction of the MPEG-21 standard *DIADescriptionType* and includes four main elements: *Resource-Properties*, *CATProperties*, *ConversionProperties* and *UsageEnvProperties*, which correspond to the four main groups of properties that we have identified. Those groups are further discussed in [14]. Each one of those elements contains XPath expressions that point out to the specific keys and values of the properties within the MPEG-7, MPEG-21 and CAT Capabilities description documents. Listing 3 shows the more relevant parts of a *Properties Repository* example (available online in *pr.xml*). This document contains all the XPath expressions in use currently within CAIN.

This set of properties can be used in different adaptation scenarios where different DIs, CAT Capabilities and UED documents (that comply with the prearranged schema) can be used without making changes in the ADM or AEM algorithms.

```
<dia:DIA xmlns="urn:gti:cain-properties-repos"
         xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-NS"
 <dia:Description xsi:type="PropertiesReposType">
  <ResourceProperties required="true">
   <Property name="id" required="true"  xpath="/@id"/>
   ..........................................
  </ResourceProperties>
  <CATProperties required="true">
  ..........................................
  </CATProperties>
  <ConversionProperties required="true">
   <Property name="lossy" required="false"  xpath="/Lossy"/>
   ..........................................
  </ConversionProperties>
  <UsageEnvProperties required="true">
  ..........................................
  </UsageEnvProperties>
 </dia:Description>
</dia:DIA>
```

Listing 3: *Properties Repository Description* example

7 Conclusions

In this document we propose the distinction between the *UED* - capable of describing different usage environments - and the *ARC Description* - that selects one of those environments before performing the adaptation -. In addition, we have emphasized the different kinds of multimedia composition and adaptation levels that ought to be contemplated by an all-purpose adaptation engine. CAIN is a multimedia adaptation engine that can be integrated within large-scale multimedia systems with the purpose of providing multimedia adaptations services. It has been successfully put into practice in the aceMedia and MESH projects that are referenced in the acknowledgments of this paper. In order to deal with the diversity, CAIN proposes two instruments: (1) the wide variety of multimedia description formats that can be dealt with the use of the CM (see section 0), so that the external multimedia representation is transformed to MPEG-21 DIs before performing the adaptation. The external representation could be recovered again after adaptation. (2) The large quantity of multimedia conversions that could be used to adapt media makes it unfeasible to implement all of them, so we propose the use of pluggable CATs. Their functionality is described in the CATs Capabilities, so that they can be used as soon as they are installed in the adaptation engine. Since each CAT could be capable of performing several conversions, we propose to divide the *CAT Capabilities Description* into *Conversion Descriptions*. In addition, in section 0 we proposed the use of a *Properties Repository* that contains the set of properties that are going to be used by the ADM to make decisions. Assuming that those *CAT Capabilities Repository* and *Properties Repository* are provided, this paper demonstrates that the automatic decision of the adaptation to perform can be addressed more effectively and efficiently.

References

[1] Ian, S., Burnett, F., Pereira, R.V., de Walle, R.: The MPEG-21 Book. John Wiley and Sons, Chichester (2006)

[2] MPEG-21 Part 2: Digital Item Description, TR, ISO/IEC 21000-2 (2005)

[3] MPEG-21 Part 7: Digital Item Adaptation, TR, ISO/IEC 21000-7 (2004)

[4] Martínez, J.M., Valdés, V., Bescós, J., Herranz, L.: Introducing CAIN: A metadata-driven content adaptation manager integrating heterogeneous content adaptation tools. In: Proceedings of WIAMIS 2005, Montreux, France (2005)

[5] López, F., Martinez, J.M.: Multimedia Content Adaptation Modelled as a Constraints Matching Problem with Optimisation. In: Proceedings of the International Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS 2007, Santorini, Greece, June 2007, pp. 82–85 (2007) ISBN 0-7695-2818-X

[6] Jannach, D., Leopold, K., Timmerer, C., Hellwagner, H.: A knowledge-based framework for multimedia adaptation. International Journal on Applied Intelligence 2(24), 109–125 (2006)

[7] Prangl, M., Szkaliczki, T., Hellwagner, H.: A Framework for Utility-Based Multimedia Adaptation. IEEE Transactions on Circuits and Systems for Video Technology 17, 719–728 (2007)

[8] Köhncke, B., Balke, W.T.: Preference-driven personalization for flexible digital item adaptation. Multimedia Systems 13(2), 119–130 (2007)

[9] International Press Telecommunication Council. NewsML 2.0 Specification. Technical Report (2007)

[10] MPEG-7 Part 5 Multimedia Content description Interface, Technical report, ISO/IEC 15938 (2003)

[11] Asadi, M.K.: Multimedia Content Adaptation with MPEG-21. PhD. thesis, ENST Paris (2005)

[12] XML Path Language (XPath) Version 1.0, Technical report, WWW Consortium (W3C) (November 1999)

[13] De Zutter, S., Van de Walle, R.: Enhanced Quality of Experience in Heterogeneous Environments from a Content Author's Perspective. In: Sixth FirW PhD Symposium, Ghent University, November 30 (2005)

[14] López, F., Jannach, D., Martínez, J.M., Timmerer, C., Hellwagner, H., García, N.: Multimedia Adaptation Decisions Modelled as Non-Deterministic Operations. In: WIAMIS 2008, Klagenfurt (May 2008)

[15] Valdés, V., Martínez, J.M.: Content Adaptation Tools in the CAIN framework. In: Atzori, L., Giusto, D.D., Leonardi, R., Pereira, F. (eds.) VLBV 2005. LNCS, vol. 3893, pp. 9–15. Springer, Heidelberg (2006)

Mobile Museum Guide Based on Fast SIFT Recognition

Boris Ruf[1], Effrosyni Kokiopoulou[1], and Marcin Detyniecki[2]

[1] Ecole Polytechnique Fédérale de Lausanne (EPFL)
{boris.ruf,effrosyni.kokiopoulou}@epfl.ch
[2] Laboratoire d'Informatique de Paris 6 (LIP6)
marcin.detyniecki@lip6.fr

Abstract. This article explores the feasibility of a market-ready, mobile pattern recognition system based on the latest findings in the field of object recognition and currently available hardware and network technology. More precisely, an innovative, mobile museum guide system is presented, which enables camera phones to recognize paintings in art galleries.

After careful examination, the algorithms Scale-Invariant Feature Transform (SIFT) and Speeded Up Robust Features (SURF) were found most promising for this goal. Consequently, both have been integrated in a fully implemented prototype system and their performance has been thoroughly evaluated under realistic conditions.

In order to speed up the matching process for finding the corresponding sample in the feature database, an approximation to Nearest Neighbor Search was investigated. The k-means based clustering approach was found to significantly improve the computational time.

1 Introduction

1.1 Motivation

Worldwide, sales of camera phones are skyrocketing. Almost every new cellphone purchased today is equipped with a built-in camera, and camera phones are projected to outsell digital standalone cameras within a few years. The Gartner Group estimates that in 2006 nearly 460 million camera phones were shipped and it forecasts that number to hit one billion devices by 2010 [1].

Cellphones have clearly evolved beyond mere conversational communication devices to ubiquitous imaging devices that support various forms of multimedia. This prevalence, coinciding with rapidly advancing communication infrastructures, initiated a growing interest in the application of image recognition on mobile devices. Using them as interactive user interfaces and image sensors has the great potential to augment the user's reality.

Several applications have already been envisioned such as bar code scanners [2], image-based object search [3] and an urban navigation system [4].

The domain this project deals with is the appealing idea of an enhanced museum tour guide. Today, museums and art galleries usually provide visitors either

M. Detyniecki, U. Leiner, and A. Nürnberger (Eds.): AMR 2008, LNCS 5811, pp. 170–183, 2010.

with paper booklets or with audio guides providing an contrived identification of system. The prototype presented here enables a camera phone to act as a museum guide: the user points with his camera phone to the painting of interest and takes a picture. Image processing technology recognizes the input picture and provides multi-modal, context-sensitive information regarding the identified painting. Details such as title, artist, historical context, critical review can be easily communicated to the visitor in the language of his choice. Such an augmented reality application could assist to appreciate art more deeply and also make it more accessible to everyone.

Using cell phones as a platform for personal museum guides would have several advantages over current audio guide systems: the interaction of taking a snapshot is found more intuitive than finding an object's number and typing it into the device. Moreover, the identification can be performed not only for the global painting, but also for details. For instance particular faces or sub-scenes of large painting or frescoes can, if the description is available, be identified.

Finally from an economical point of view, either museum operators profit by significantly reducing maintenance and specific infrastructure costs or tourist operators can develop their own products, since the visitor can use his own mobile device.

1.2 Problem Statement

Object recognition is still an open problem in computer vision, and the reasons for this are numerous. Images may be subject to variations in point of view, illumination and sharpness; different camera characteristics can also be an issue. Moreover, the museum environment has some unique properties: indoor lighting in museums can be insufficient and museum rules may prohibit using a flash. Reflection of security glass which protects pieces of art is another challenge. Camera phones still tend to have cheap lenses that produce noisy photographs of poor quality. As cell phones are not primarily designed for taking pictures they are more difficult to hold steady which in turn increases the likelihood of camera shake. In a crowded museum paintings might be partly occluded by other visitors or even cropped if the piece of art is too vast to be captured at once. Also, more than one painting may appear on the image if the paintings have been arranged close together. Frames can vary from bold, rectangular ones to subtle, oval ones and cast significant, shadowed regions. Both the shape and shadows of the frame complicate a possible segmentation of the painting incredibly. More difficulties become obvious when considering the content of the painting: the uniqueness of features is reduced as paintings from the same epoch show recurring styles and similar color schemes. In fact, in the case of studies, whole patches of some paintings can be found repeated in other paintings.

The aim of this work was to overcome these problems in a mobile real-life image matching application.

Most systems presented in related work in mobile visual communication have actually been simulated on desktop PCs. This project firmly intended to deploy the client software on a real hand-held device and evaluate its handling under

the most realistic conditions possible. For the same reason, a large database and many test samples were chosen. These requirements bear additional challenges to the implementation.

1.3 Related Work

Object recognition. Two major families of methods have evolved in the field of object recognition. The holistic *global* feature approach handles the entire image as one entity, while the *local* feature approach selects distinctive regions in the image.

The most obvious *global* features are color histograms. A recognition system based on color histograms was presented by Swain and Ballard [5] in 1991. Face recognition is a well explored domain which often relies on global features [6], [7]. Some of the most popular algorithms in this field include Eigenfaces [8], which uses Principal Component Analysis (PCA), and Fisherfaces [9], which adopts Fisher Linear Discriminant Analysis (FLD). PCA methodologies select a dimensionality reducing linear projection which models the data by maximizing its scatter. FLD techniques attempt to improve reliability for classification problems by taking into account the classes and maximizing the ratio between the intra-class and the extra-class.

In 1988, Harris and Plessey introduced the Harris corner detector [10] to find local interest points. Mohr et al. later applied this concept to locate invariant features and matched them against a large database [11]. In 1996, van Gool introduced generalized color moments that represent the shape and the intensities of different color channels in a local region [12]. In 1999, Lowe presented the Scale-Invariant Feature Transform algorithm (SIFT) which achieved scale invariance using local extrema detected in Gauss-filtered difference images for object recognition [13]. In 2002, Siggelkow showed methods to use local feature histograms for content-based image retrieval [14]. In the same year Schaffalitzky and Zisserman investigated how a combination of image invariants, covariants, and multiple view relations can be used for efficient multiple view matching [15]. Mikolajczyk and Schmid used the differential descriptors to approximate a point neighborhood [16]. In 2004, Till Quack et al. introduced Cortina, a large-scale image retrieval system for images of the Web based on low-level MPEG-7 visual features and indexed keywords as additional high-level feature.[17] Combined with association rule mining this concept sucessfully improved the quality of the search results.

Experimental museum guide systems. In 2002, Kusonoki et al. presented a location-aware sensing board for kids which gives visual and auditory feedback to attract users' interests. Interactive museum tour-guide robots have been proposed by Burgard [18] in 1998 and Thrun [19] in 2000.

In January 2005, Adriano Alberti et al. described an augmented reality system using video see-through technology that provides contextual information for details of one painting [20]. The system is trained with a large number of

Fig. 1. High-level architecture of the prototype system

synthetically generated images. The recognition process utilizes a set of multidimensional receptive field histograms that represent features such as hue, edginess and luminance.

An Interactive Museum Guide [21] that is capable of recognizing objects in the Swiss National Museum in Zurich was proposed by Herbert Bay et al. in September 2005. In order to reduce the search space, Bluetooth emitters were installed on site. Objects are recognized with an approximated SIFT algorithm.

In October 2005 Erich Bruns et al. from the Bauhaus University in Weimar presented the PhoneGuide [22]. Two-layer neural networks are used in combination with Bluetooth emitters and trained directly on the mobile phone. All computation for object recognition is carried out on the device.

The French-Singapore IPAL Joint Lab presented in July 2007 the Snap2Tell prototype [23] which recognizes tourist attractions and provides multi-modal descriptions. Scenes are recognized by distinguishing local discriminative patches described by color and edge information. As discriminative classifiers Support Vector Machines (SVMs) are used. The reference database contains a notable number of images per object and GPS was evaluated as additional feature.

2 System Description

2.1 Architecture

A PDA with integrated camera and Internet connection was enabled to act as a universal museum guide for paintings in art galleries. In contrast to conventional audio museum guides or booklets, objects are selected by simply taking a picture of them.

The major advantage of the system presented here over other experimental systems that have been proposed in previous work on the same subject is that

it does not depend on additional infrastructure on site. Neither barcode labels nor extra hardware such as Bluetooth emitters need to be set up.

The architecture follows the classical server-client approach: the client only acts as periphery which acquires and sends sample data and eventually receives the results. No additional computation such as feature extraction is executed on the client. This decision has been taken for several reasons: the CPU of mobile clients is generally very slow and running the feature extraction on the mobile client might result in unbearably long waiting times for the user. Also, the recognition performance is good even at low resolution. Transmitting scaled down images of small data size is sufficient for successful operation of the system.

Mobile clients have been developed for Windows Mobile and the Android operating system by Google. Furthermore, a web browser-based interface was implemented to enable access to the painting recognition system through the Internet.

A very basic, high-level description of the architecture of the system is shown in Figure 1.

2.2 Hardware

All images were captured with a HP iPAQ hw6900 handheld device and have an original resolution of 1280×1024. The experiments were conducted on a virtual private server equipped with an Intel(R) Xeon(TM) CPU 2.80GHz, 384 MB RAM and Debian Linux 3.1.

3 Feature Extraction

After evaluating several methods for object recognition, Scale-Invariant Feature Transform (SIFT), conceived by David G. Lowe et al. in 1999, and Speeded Up Robust Features (SURF), introduced by Herbert Bay et al. in 2006, were identified as most appropriate for museum-inherent challenges. Both are robust regarding scale, lighting and perspective distortion. But, again, their greatest benefit is the use of local features. When employing algorithms with global features, the objects of interest first need to be clipped away from any background. In this case, the reference samples in the database show only the painting with neither frame nor background. The test samples taken in the museum, however, include parts of the environment: often, paintings are surrounded by massive frames. The wall does not always contrast clearly with the piece of art. Visitors or objects besides the painting of interest may appear on the photos. If the image was taken from a distance, the size of the painting proportional to the total image size can vary significantly. Detecting the painting becomes particularly challenging when it is surrounded by shadowed regions or if the frame is of unusual shape like oval. Segmentation techniques for clipping away the background before classifying the foreground are expensive and prone to failure due to these factors. This step can be skipped when using local features.

3.1 Scale-Invariant Feature Transform (SIFT)

The Scale-Invariant Feature Transform (SIFT) [13] algorithm provides a robust method for extracting distinctive features from images that are invariant to rotation, scale and distortion. In order to identify invariant keypoints that can be repeatably found in multiple views of varying scale and rotation, local extrema are detected in Gauss-filtered difference images. Stability of the extrema is further ensured by rejecting keypoints with low contrast, and keypoints localized along edges. As keypoint descriptor, an orientation histogram is computed for the area around the keypoint location. Gradient magnitude and the weight of a Gaussian window originating at the keypoint add to the value of each sample point within the considered region.

3.2 Speeded Up Robust Features (SURF)

The Speeded Up Robust Features (SURF) [24] algorithm is a variation of the SIFT algorithm. Its major differences include a Hessian matrix-based measure as an interest point detector and approximated Gaussian second order derivatives using box type convolution filters. Here, the use of integral images [25] enables rapid implementation.

4 Matching Process

4.1 Nearest Neighbor Search (NNS)

A straightforward approach to find the match of a sample keypoint within the reference keypoints is Nearest Neighbor Search (NNS). Here, the closest candidate measured by Euclidean distance is found by linearly iterating over all reference keypoints in no particular order. This method results in finding the exact nearest neighbor to the sample keypoint. Two keypoints are considered a match if the distance between them is closer than 0.6 times the distance of the second nearest neighbor [26] [27]. However, for large data sets and high-dimensional spaces this is an inefficient approach due to the time complexity of $O(N \cdot d)$ where N is the number reference keypoints and d is the dimensionality of a keypoint vector.

4.2 Best-Bin-First (BBF)

Jeffrey S. Beis and David G. Lowe proposed an approximation to NNS called Best-Bin-First (BBF) [28].

The index structure used to store the keypoints is a k-d tree. When creating the tree, the data set is recursively subdivided into even groups on iterating dimensions. At each split, the keypoint which contains the median becomes a new internal node. This step is repeated for the children at the next dimension on the elements of the subgroups. The resulting tree is balanced and binary with a depth $d = \lceil \log_2 N \rceil$ where N is the number reference keypoints.

In order to find the nearest neighbor of a sample keypoint, the tree is first traversed to locate the bin which contains the sample keypoint. The algorithm backtracks from this bin, considering each node along the way for comparison. If the distance to a node is greater than the shortest distance found so far, the subtree of this node can be ignored.

According to [28], with x=200, this approximation provides a 2 order of magnitude speed-up over exhaustive NNS, and still returns the correct nearest neighbor more than 95% of the time. In our case, however, preliminary tests on a subset of the data revealed unacceptable loss of performance.

4.3 K-Means Based Tree

A different tree-based clustering approach adopted from the paper "Tree-Based Pursuit: Algorithm and Properties" by Jost et al. [29] was evaluated.

Here, clustering is achieved based on the Euclidean distance between the vectors. The k-means algorithm [30] is used to cluster the data set into k subgroups. The centroids found become internal nodes of the tree. Recursively, the clusters are subdivided in the same manner until they consist of less than k elements. Once this state is reached, the elements of the cluster become children of their centroid node and leaf nodes of the tree. The resulting tree is not balanced and its shape highly depends on the data set, the quality of the initial centers and the value of k. The matching process of a new element breaks down to tree traversal from the root node to the bottom of the tree always choosing the node of lowest Euclidean distance. The leaf node is then considered the nearest neighbor.

5 Experimental Results

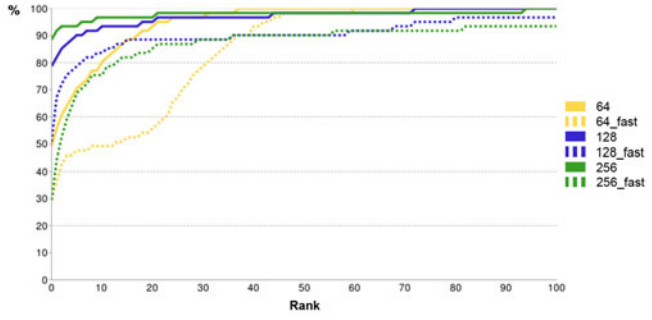
5.1 Setup

Training sample data has been extracted from the online archive Web Gallery of Art [31]. More precisely, all 1,002 works available from the Louvre Museum were considered in the experiment. Each reference painting is represented by one sample. The paintings from the online source have been digitalized without frame.

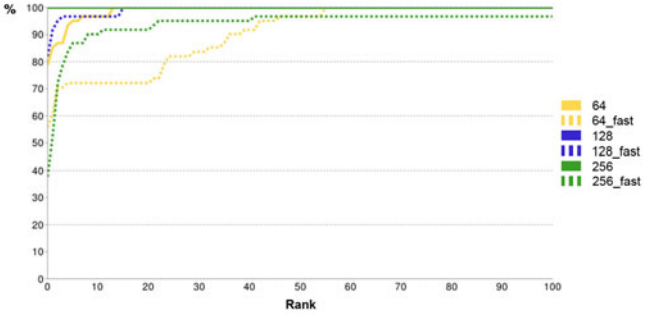
The test sample data consists of photo series of 48 paintings taken in the Louvre Museum (in total 200 images). Four different types of perspective have been considered to stress test the algorithms and also evaluate their robustness under extreme perspectives: frontal, left, right, distant.

In order to remove noise, the images have been converted to gray-level representation. To evaluate the correlation between resolution and performance of the algorithms, the images have been downsampled to 4 different resolutions: 512×410, 256×205, 128×103, 64×51.

Cumulative Match Characteristic (CMC) curves summarize the accuracy of a recognition system: for each test sample, its rank is determined by finding the position of the hypothesis for the desired, correct reference sample on a sorted



(a) All perspectives



(b) Frontal

Fig. 2. Performance comparison for approximated NNS

list of all hypotheses constructed for this sample. Ideally, the rank is 0. In this case, the hypothesis for the correct painting also received the most votes, and the sample could be identified successfully. The CMC chart integrates these results and depicts the probabilities of identification for all ranges of ranks.

5.2 SIFT vs. SURF

Figure 3 shows CMC curves for the results of linear matching using NNS, grouped by perspective. The charts on the left side result from employing the SIFT algorithm, corresponding curves for the SURF algorithm can be found on the opposite side.

It can be seen that higher resolution does not necessarily correspond to better recognition rate. For the frontal perspective, even very low resolution yields satisfying results.

5.3 Approximated SIFT

The k-means based clustering approach has been coined *SIFT_fast* and was implemented with $k = 15$.

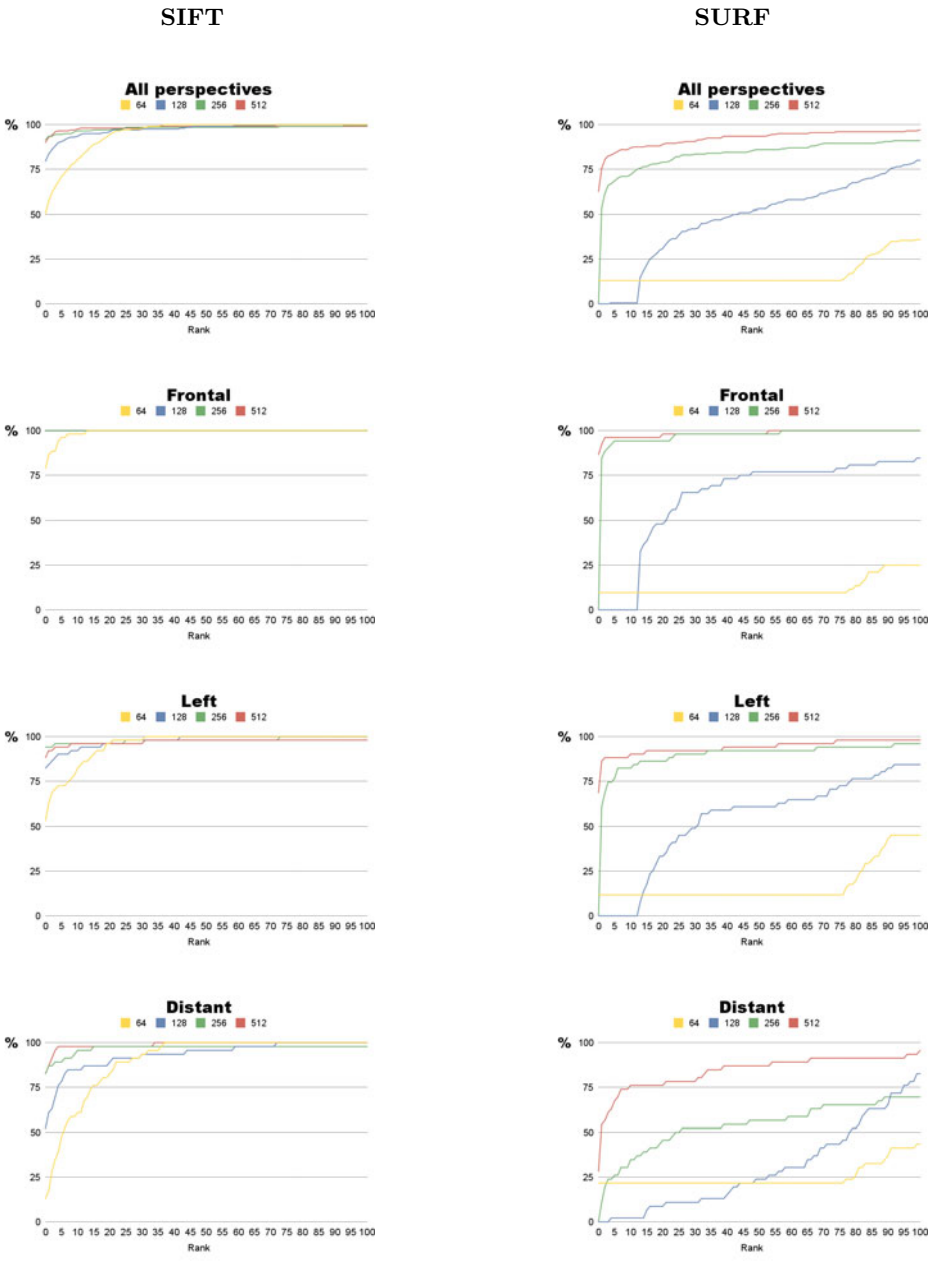


Fig. 3. CMC curves illustrate the probabilities of identification for SIFT and SURF

Figure 2 shows CMC curves for the experiment results using a linear matching approach (solid lines) and the approximated k-means tree approach introduced in Section 4.3 (dashed lines). Performance losses compared to the exhaustive approach are obvious, however, for a real-time application that deals mainly with frontal views (as the museum guide in this work does), the algorithm *SIFT_fast* for resolution 128 offers an acceptable trade-off between speed and performance.

5.4 Processing Time

The table in Figure 4 lists the average times of the matching process depending on algorithm and resolution; Figure 5 clarifies the proportions graphically.

The runtime computational complexity of SURF is lower for all resolutions. This is due to the fact that SURF descriptor vectors are of dimension 64 in contrast to 128 components contained in the descriptor vectors of SIFT. However, the median of the number of keypoints is lower, too, which has direct influence on the recognition performance: SURF is inferior to SIFT in any experiment.

The large time increase of the conventional SIFT algorithm between resolution 128 and 256 can be explained by the huge variance of keypoints of this algorithm. The variance of SURF keypoints is much smaller in comparison. This is beneficial as it makes the runtime of the matching process more predictable.

The gain of time achieved when matching SIFT keypoints using a k-means tree compared to linear NNS is significant: with resolution 128, the approximated approach takes 45 seconds instead of about 306 seconds using linear NNS matching. The downside clearly is a loss of performance as shown in Figure 2.

	64	128	256
SIFT	144.25	305.73	1440.36
SURF	78.54	95.56	198.57
SIFT_fast	13.85	44.65	150.68

Fig. 4. Table of average processing times in seconds

6 Discussion

In general, the evaluation reveals that the SIFT algorithm outperforms the SURF algorithm for any resolution considered. However, the runtime computational complexity of SURF is lower due to the fact that SURF descriptor vectors are of lower dimension than descriptor vectors of SIFT. The variance of the number of keypoints found with SURF is much smaller compared to the distribution of SIFT keys. This is advantageous as it makes the runtime of the matching process more predictable. However, the median is lower, too, which has direct influence on the recognition performance. In fact, the strength of the SURF algorithm only becomes apparent at the highest resolution of 512 $\times$ 410 tested in the experiments. SIFT features, on the other hand, show sufficient distinctive power

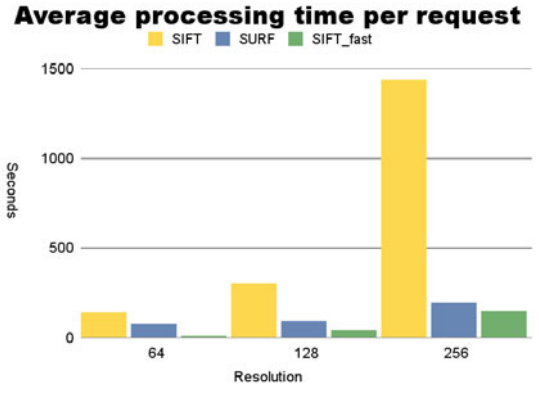


Fig. 5. Average processing times visualized for comparison

even for images of significantly lower resolution than used in the experiment section of the SIFT paper (600 × 315). Our experiments show that input images of 128 × 103 already deliver reasonable performance.

These findings, and the fact that programming on mobile platforms is rather cumbersome, implies an architecture on which the feature extraction part is done on the server.

Analysis of the experimental data also clearly showed that perspective distortion is still an issue. However, for an application as described in this project, it is acceptable to assume a frontal perspective and to choose rather low resolution parameters in order to strike a balance between efficiency and accuracy.

Moreover, clustering methods which approximate the conventional Nearest Neighbor Search are an important extension to a recognition system, in particular to a real-life application such as this one. In fact, they enormously speed up the response time. The tests show that the k-means based tree approach provides an acceptable trade-off between performance loss and gain of time.

7 Conclusion

The results presented in this article demonstrate the feasibility of a market-ready mobile pattern recognition system in the form of a universal museum guide. Several prototype clients were fully implemented and have been subject to thorough evaluation under realistic conditions.

Our tests showed the advantages of an architecture where the feature extraction part is done on the server. Such a setup requires uploading images and favors low resolutions, as this decreases the response time. Although the SURF algorithm is faster than the SIFT one, for low resolution images SURF's performance is unacceptable.

Our tests further showed that methods which approximate the conventional Nearest Neighbor Search can also reduce response times. The k-means based tree

approach provided an acceptable trade-off between performance loss and gain of time.

Finally, based on this study, we conclude that a combination of client-server architecture, the use of a SIFT algorithm with a resolution of 128 × 103, combined with the k-means based tree approach is most appropriate for deployment.

The extension of the presented framework to standard representations such as MPEG-7 would require deeper examination and remains as interesting objective for the future.

Acknowledgments

Many thanks to Prof. Pascal Frossard for providing valuable feedback on this article. Special thanks to Dr. Emil Krén and Dr. Dániel Marx without their generous permission to use the entire image data from Web Gallery of Art [31] the experiments would not have been possible.

References

1. Group, G.: 2006 Press Releases. November 2 (2006), http://www.gartner.com/it/page.jsp?id=498310
2. Rohs, M., Gfeller, B.: Using camera-equipped mobile phones for interacting with real-world objects. In: Advances in Pervasive Computing, pp. 265–271. Austrian Computer Society (OCG), Austria (2004)
3. Yeh, T., Grauman, K., Tollmar, K., Darrell, T.: A picture is worth a thousand keywords: image-based object search on a mobile platform. In: CHI 2005: CHI 2005 extended abstracts on Human factors in computing systems, pp. 2025–2028. ACM, New York (2005)
4. Robertsone, D., Cipolla, R.: An image-based system for urban navigation. In: The 15th British Machine Vision Conference (BMVC 2004), pp. 819–828 (2004)
5. Swain, M.J., Ballard, D.H.: Color indexing, vol. 7(1), pp. 11–32. Kluwer Academic Publishers, Hingham (1991)
6. Chellappa, R., Wilson, C.L., Sirohey, S.: Human and machine recognition of faces: A survey. Proceedings of the IEEE 83(5), 705–740 (1995)
7. Samal, A., Iyengar, P.A.: Automatic recognition and analysis of human faces and facial expressions: a survey. Pattern Recogn. 25(1), 65–77 (1992)
8. Turk, M., Pentland, A.: Eigenfaces for recognition. CogNeuro. 3(1), 71–96 (1991)
9. Belhumeur, P., Hespanha, J., Kriegman, D.: Eigenfaces vs. Fisherfaces: recognition using class specific linear projection. Transactions on Pattern Analysis and Machine Intelligence 19(7), 711–720 (1997)
10. Harris, C., Stephens, M.: A combined corner and edge detection. In: Proceedings of The Fourth Alvey Vision Conference, pp. 147–151 (1988), http://www.csse.uwa.edu.au/ pk/research/matlabfns/Spatial/Docs/Harris/A_Combined_Corner_and_Edge_Detector.pdf
11. Schmid, C., Mohr, R.: Local grayvalue invariants for image retrieval. IEEE Trans. Pattern Anal. Mach. Intell. 19(5), 530–535 (1997)
12. Gool, L.J.V., Moons, T., Ungureanu, D.: Affine/ photometric invariants for planar intensity patterns. In: Buxton, B.F., Cipolla, R. (eds.) ECCV 1996. LNCS, vol. 1064, pp. 642–651. Springer, Heidelberg (1996)

13. Lowe, D.: Object recognition from local scale-invariant features. In: The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999, vol. 2, pp. 1150–1157 (1999)
14. Siggelkow, S.: Feature histograms for content-based image retrieval. Ph.D. dissertation, University of Freiburg, Institute for Computer Science (2002)
15. Schaffalitzky, F., Zisserman, A.: Multi-view matching for unordered image sets, or 'How do i organize my holiday snaps? In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 414–431. Springer, Heidelberg (2002)
16. Mikolajczyk, K., Schmid, C.: An affine invariant interest point detector. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 128–142. Springer, Heidelberg (2002)
17. Quack, T., Monich, U., Thiele, L., Manjunath, B.: Cortina: A system for large-scale, content-based web image retrieval. In: ACM Multimedia 2004 (October 2004)
18. Burgard, W., Cremers, A., Fox, D., Hähnel, D., Lakemeyer, G., Schulz, D., Steiner, W., Thrun, S.: The interactive museum tour-guide robot. In: Proc. of the Fifteenth National Conference on Artificial Intelligence, AAAI 1998 (1998)
19. Thrun, S., Beetz, M., Bennewitz, M., Burgard, W., Cremers, A., Dellaert, F., Fox, D., Hähnel, D., Rosenberg, C., Roy, N., Schulte, J., Schulz, D.: Probabilistic algorithms and the interactive museum tour-guide robot Minerva. International Journal of Robotics Research 19(11), 972–999 (2000)
20. Albertini, A., Brunelli, R., Stock, O., Zancanaro, M.: Communicating user's focus of attention by image processing as input for a mobile museum guide. In: IUI 2005: Proceedings of the 10th international conference on Intelligent user interfaces, pp. 299–301. ACM, New York (2005)
21. Bay, H., Fasel, B., Gool, L.V.: Interactive museum guide: Fast and robust recognition of museum objects. In: Proceedings of the first international workshop on mobile vision (2006)
22. Bruns, E., Brombach, B., Zeidler, T., Bimber, O.: Enabling mobile phones to support large-scale museum guidance. IEEE MultiMedia 14(2), 16–25 (2007)
23. Lim, J.-H., Li, Y., You, Y., Chevallet, J.-P.: Scene recognition with camera phones for tourist information access. In: IEEE International Conference on Multimedia and Expo., 2007, July 2-5, pp. 100–103 (2007)
24. Bay, H., Tuytelaars, T., Van Gool, L.: SURF: Speeded Up Robust Features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 404–417. Springer, Heidelberg (2006)
25. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: CVPR, vol. 1, pp. I–511 – I–518 (2001)
26. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(10), 1615–1630 (2005)
27. Baumberg, A.: Reliable feature matching across widely separated views. In: Proceedings of Computer Vision and Pattern Recognition, 2000, vol. 1, pp. 774–781 (2000)
28. Beis, J., Lowe, D.: Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In: 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997. Proceedings, June 17-19, pp. 1000–1006 (1997)

29. Jost, P., Vandergheynst, P., Frossard, P.: Tree-Based Pursuit: Algorithm and Properties. IEEE Transactions on Signal Processing 54(12), 4685–4697 (2006)
30. Macqueen, J.B.: Some methods of classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
31. Kren, E., Marx, D.: Web Gallery of Art, http://www.wga.hu

Author Index