universität wien

Masterarbeit

Titel der Masterarbeit

Audio Content Identification – Fingerprinting vs. Similarity Feature Sets

angestrebter akademischer Grad

Diplom-Ingenieur (Dipl.-Ing.)

Verfasserin / Verfasser:	Gerhard Sageder
Matrikel-Nummer:	0301696
Studienkennzahl (lt. Studienblatt):	A 066 935
Studienrichtung (lt. Studienblatt):	Medieninformatik Masterstudium
Betreuer:	Univ.Prof.Dr. Wolfgang Klas, Ao. Univ.Prof.Dr. Andreas Rauber
mitbetreuende Assistenten:	Mag. Stefan Leitich, Dipl.Ing. Thomas Lidy

Wien, im Februar 2009

Among all I would like to thank my parents who gave me the opportunity to attend university at all and have always supported my work in all conceivable aspects, my girlfriend Corinna for showing me understanding for partly far too much time in front of lengthy computations and implementation efforts. Further on, I really appreciate the supervision of my thesis and the facility of working on this scientific field.

Gerhard Sageder

Abstract

The development and research of content-based music information retrieval (MIR) applications in the last years have shown that the generation of descriptions enabling the identification and classification of pieces of musical audio is a challenge that can be coped with. Due to the huge masses of digital music available and the growth of the particular databases, there are investigations of how to automatically perform tasks concerning the management of audio data.

In this thesis I will provide a general introduction of the music information retrieval techniques, especially the identification of audio material and the comparison of similarity-based approaches with content-based fingerprint technology. On the one hand, similarity retrieval systems try to model the human auditory system in various aspects and therewith the model of perceptual similarity. On the other hand there are fingerprints or signatures which try to exactly identify music without any assessment of similarity of sound titles. To figure out the differences and consequences of using these approaches I have performed several experiments that make clear how robust and adaptable an identification system must work. Rhythm Patterns, a similarity based feature extraction scheme and FDMF, a free fingerprint algorithm have been investigated by performing 24 test cases in order to compare the principle behind. This evaluation has also been done focusing on the greatest possible accuracy. It has come out that similarity features like Rhythm Patterns are able to identify audio titles promisingly as well (i.e. up to 89.53%) in the introduced test scenarios. The proper choice of features enables that music tracks are identified at best when focusing on the highest similarity between the candidates both for varied excerpts and signal modifications.

Kurzfassung

Die Entwicklung und Erforschung von inhaltsbasierenden “Music Information Retrieval (MIR)” - Anwendungen in den letzten Jahren hat gezeigt, dass die automatische Generierung von Inhaltsbeschreibungen, die eine Identifikation oder Klassifikation von Musik oder Musikteilen ermöglichen, eine bewältigbare Aufgabe darstellt. Aufgrund der großen Massen an verfügbarer digitaler Musik und des enormen Wachstums der entsprechenden Datenbanken, werden Untersuchungen durchgeführt, die eine möglichst automatisierte Ausführung der typischen Managementprozesse von digitaler Musik ermöglichen.

In dieser Arbeit stelle ich eine allgemeine Einführung in das Gebiet des “Music Information Retrieval” vor, insbesondere die automatische Identifikation von Audiomaterial und den Vergleich von ähnlichkeitsbasierenden Ansätzen mit reinen inhaltsbasierenden “Fingerprint”-Technologien. Einerseits versuchen Systeme, den menschlichen Hörapparat bzw. die Wahrnehmung und Definition von “Ähnlichkeit” zu modellieren, um eine Klassifikation in Gruppen von verwandten Musiktiteln und im Weiteren eine Identifikation zu ermöglichen. Andererseits liegt der Fokus auf der Erstellung von Signaturen, die auf eine eindeutige Wiedererkennung abzielen ohne jede Aussage über ähnlich klingende Alternativen. In der Arbeit werden eine Reihe von Tests durchgeführt, die deutlich machen sollen, wie robust, zuverlässig und anpassbar Erkennungssysteme arbeiten sollen, wobei eine möglichst hohe Rate an richtig erkannten Musikstücken angestrebt wird. Dafür werden zwei Algorithmen, Rhythm Patterns, ein ähnlichkeitsbasierter Ansatz, und FDMF, ein frei verfügbarer Fingerprint-Extraktionsalgorithmus mittels 24 durchgeführten Testfällen gegenübergestellt, um die Arbeitsweisen der Verfahren zu vergleichen. Diese Untersuchungen zielen darauf ab, eine möglichst hohe Genauigkeit in der Wiedererkennung zu erreichen. Ähnlichkeitsbasierte Ansätze wie Rhythm Patterns erreichen

bei der Identifikation Wiedererkennungsraten bis zu 89.53% und übertreffen in den durchgeführten Testszenarien somit den untersuchten Fingerprint-Ansatz deutlich. Eine sorgfältige Auswahl relevanter Features, die zur Berechnung von Ähnlichkeit herangezogen werden, führen zu äußerst vielversprechenden Ergebnissen sowohl bei variierten Ausschnitten der Musikstücke als auch nach erheblichen Signalveränderungen.

Contents

Chapter 1

Introduction

In this chapter I would like to provide an introduction into the field of music identification and the lack of possibilities for analog devices. Furthermore I will give an overview about common - already existing - applications, and in what sense they can give an edge to end users or customers.

1.1 Motivation

In the last years, the rapid increase of interest in digital music, the storage and organization of big libraries and - above all - the availability of music in that form itself, has led to the development of systems which satisfy lots of demands of the respective community. Besides the conventional way of storing music - mostly in a compressed file format - on a hard disk drive on a personal computer or portable device, more and more network based music delivery techniques have come into being. There is a huge amount of available internet radio stations which provide their audio data via standardized streaming protocols and formats. In many cases, these do not support proper meta data delivery or the client side streaming player cannot process them efficiently. Moreover this feature is in fact unable to be provided in many cases. Consider that music is still often recorded or transformed from analog sources. These sources cover instrumental live recordings as well as analog mediums. A file format is capable of offering included meta data automatically which can be added to the audio stream. Analog - non semantically enhanced - data does not know about its

contents and has to be annotated before in order to provide semantics like the digital match. Nowadays it is recommended that this process is automated and need not be done by hand. If we start from the assumption that the same piece of music is also available in a digital form which in fact has already been associated with additional meta data, then there must be a programmatic way of matching the content of these basically different, but concerning the contents equivalent versions.

Another related issue is the matter that there is still a huge amount of people who desire the existence and maintenance of analog mediums - especially vinyl records. These are said to be more precise in various aspects, (e.g. frequency bands) and in general more pleasant for active listeners or in settings of live performance. It is not likely that this community intends to change its habits and totally abstains from old fashioned vinyl records. But what actually happens is, that the audio signal is often recorded, saved and encoded to a digital format which then is published over network-based services or storage mediums. If we can manage the process of identifying the audio data by comparing them to already verified digital data, we could ensure that some of the benefits of digitalization can be achieved for analog data via the specified workaround.

1.2 Applications

In times of sharing music over the internet, storing and listening to downloaded tracks on a personal MP3-player or consuming music from one of the huge amounts of broadcasting stations, many use cases for audio fingerprinting have arisen and have led to immediate applications which are intended to fulfil the requirements of both music content consumers and producers.

One big application refers to the broadcasting industry. The large number of radio stations complicates the choice of selecting one appropriate program. There are several projects which follow up the automatic organization and categorization of these stations for facilitating a users decision and generating individual profiles [LR06]. On the other hand, the industry is interested in applications which monitor users demands, transmitted audio tracks and - above all - the prohibition of unauthorized

copies of broad-casted material that is protected by copyright laws [Cer07]. Furthermore the realm of radio and television offers connection points to applications that can automatically track advertisements or even replace them [BC06] [Cer07]. This is very important for companies to ensure that their ad spot is really broadcasted for and at the right time.

Another interesting issue is the automatic track-list generation and categorization of real time analog audio signals. In settings of live studio performance or public shows, artists still have to provide their play-list manually. If this step can be automated too, the interested parties would be be able to move together in more tighter way.

According to a consumers point of view, one can find much more applications. Imagine that users would be able to identify music titles which they are listening to at the moment independently of the corresponding host. Consumer electronic devices can have software integrated that performs an automatic identification by computing a fingerprint and comparing it to a built-in database which can be updated either by hand or over a wireless connection. These devices can be MP3-players, handheld computers, stationary hi-fi equipment, mobile phones and many more [HK02].

1.3 Outline

This thesis is subdivided into seven main chapters that cover the following aspects. Theses sections provide a deep understanding for both dedicated fingerprint approaches and audio similarity measurement which are both used for an audio identification system based on the content only.

(2) Related Work

A description of related articles and scientific results that have already been achieved in previous studies. Scientific works, the contents of which has to be taken into account, are introduced as well as the relevance to the research field.

(3) Audio Content Identification

This chapter deals with general audio identification and fingerprint techniques, common problems and the contrast to similarity based approaches.

(4) Audio Similarity Feature Sets

The general procedure of extracting features from audio signals and the comparison to other elements using distance measures is illustrated in this section as well as considerations for evaluation and quality assurance.

(5) Experimental Results

In this chapter, the whole experimental part is described in detail providing numerical and comparable results. The test series include 24 test cases which cover input length considerations as well as evaluation techniques and signal modifications.

(6) Implementation Details

The results of the JAVA implementation with architecture details is illustrated in this part.

(7) Conclusion and Outlook

Finally, conclusions and findings are summarized and an outlook for future ideas is given.

Chapter 2

Related and Fundamental Work

In this chapter, related articles and scientific works of community members are introduced to provide background information and the respective context.

The scientific field of content-based music information retrieval has become a very popular and promising discipline that enables a lot of capabilities for humans interested in music. The management of huge databases as well as the computational extraction schemes, procedures for classification and identification approaches have led to the requirement of large-scale community discussions. Applications have shown that the identification of pieces of music can work very robustly and therewith nearly independently of the transmission format and quality. Haitsma and Kalker are the authors of one of the most well-known papers in the scientific field concerning the automatic identification of audio signals via robust hashes and sub-fingerprint database strategy. (cp. [HK02]). This approach has become a common guideline describing how to design a reliable feature extraction system for generating signatures.

There is a variety of descriptors that try to identify music. Betser, Collen and Rault (cp. [BCR07]) present a new descriptor which is based on sinusoidal modeling for jingle detection. It is robust against common distortions and signal deterioration as well as non-random noise additions like speech overlay. The issue that some signal which is not relevant for and potentially harmful to the identification process is added to the original piece of music is a challenging task. This leads to requirement of not just identifying the original data or their derivates due to physical reasons, format

changes, bitrate decrease, etc., but of recognizing remix versions or severe content alterations. (cp. [CS07], [CS06], [DL05])

The classification of audio material based on pre-defined terms like the genre is another popular problem that has to be solved. A proper combination of various features concerning the musical surface, rhythm and others can build vector representations which are used for the association of audio tracks to genre-terms (cp. [TEC02]). This approach has become the main idea behind selecting representative features which constitute a comparable semantic description for the underlying music track.

Details of the two main approaches (similarity-based vs. fingerprinting) as well as related articles are given by Chapter 3 and Chapter 4.

Chapter 3

Audio Content Identification

In this chapter I will give an overview about the meanings of audio content as well as the differences between audio in terms of music and other audio signal producing sources like the human voice (Section 3.1). In Chapter 3.2, various identification approaches and techniques are introduced to outline how content identification can be achieved via fingerprinting. Section 3.3 is about matching fingerprints with database entries and measuring the distances in order to retrieve reliable results for a recognition request.

3.1 Audio Content

3.1.1 Music Information

Music information is the entirety of basic audio related informative messages which are included in a piece of music. This information is transmitted by sound waves through the air and reaches our auditory system, which filters the signal and passes the relevant parts to our brain. This in general is called acoustic perception. Thus, music contains information that is extractable by the human auditory sense.

The common human understanding of music is lying in the time domain. Music titles are played back on the time axis and have a defined length as the representation of the amplitude that is varying over the time let suppose. But this information alone does not provide features which can be used for calculations or extractions which

would be useful for creating convincing descriptions with the purpose to identify music unambiguously. Foremost the transformation of an audio signal from the time domain to the frequency domain enables the possibility to make statements about the existence and progression of periodical elements as well as pitch, frequency ranges, harmonics and many more. The most frequently used and well known Fourier Transformation performs this fundamental transformation. Equation 3.1 provides the formal definition for the DFT (Discrete Fourier Transformation) which is used for discrete digital signals.

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} nk} \qquad k = 0, \ldots, N-1. \tag{3.1}$$

... where x_n are the time-based complex numbers, N the number of values to transform and X_k the resulting Fourier transformed complex numbers - also called the Fourier Coefficients. The inversion, the transformation of a signal in the frequency domain to the time domain is given by equation 3.2. The variable denomination conforms to equation 3.1.

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{\frac{2\pi i}{N} kn} \qquad n = 0, \ldots, N-1 \tag{3.2}$$

The frequency domain allows several extractions and computational time-invariant possibilities that bring out characteristic features and representations like spectrograms (cp. Figure 3.1), energy deviations, frequency histograms and the magnitude of certain frequency ranges respectively transformations that illustrate its influences to our perception. In addition to the analysis of sinusoidal frequency and phase contents, the time discrete Short Time Fourier Transformation (STFT) performs on short single segments of a signal that is changing over time. The result is a representation of frequency lots at a specific time, commonly visualized by a time-frequency diagram.

It is used in applications that have to maintain a rough subdivision and allocation of frequency ranges to particular sections of an input signal. This is especially true for systems which realize a segmentation of different parts. (e.g. separation of

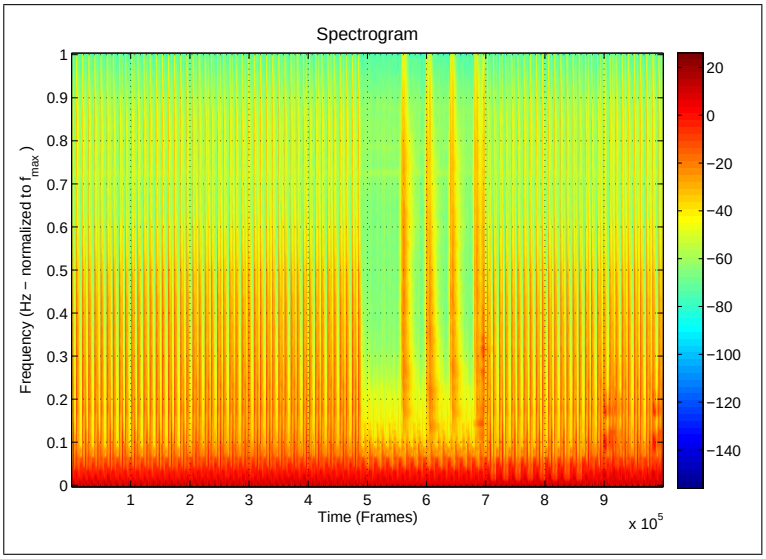

Figure 3.1: *Spectrogram example - Frequency-time representation of an audio track*

advertisement from music content on broad-casted material). [MTB+04], [HKO02], [KSWW00], [Pei07]

The Bark scale is a psycho-acoustical scale that matches frequency range intervals to a specified number. It is based on the perception of pitch of human beings with respect to the relative acoustic feeling. It considers the almost linear relation in lower frequency ranges as well as the logarithmic in higher ranges and its basic idea originates from frequency grouping and the subdivision concept as it is known from research concerning the human ear. The association to the frequency ranges is given by Table 3.1.

3.1.2 Audio Content Models

In general, there must be a pre-defined model for any identification scenario that works like the described schemes. This can include assumptions about the frequency domain or the sequence of acoustic events in the time domain.

z	Hz	z	Hz	z	Hz	z	Hz
1	$20-100$	7	$630-770$	13	$1720-2000$	19	$4400-5300$
2	$100-200$	8	$770-920$	14	$2000-2320$	20	$5300-6400$
3	$200-300$	9	$920-1080$	15	$2320-2700$	21	$6400-7700$
4	$300-400$	10	$1080-1270$	16	$2700-3150$	22	$7700-9500$
5	$400-510$	11	$1270-1480$	17	$3150-3700$	23	$9500-12000$
6	$510-630$	12	$1480-1720$	18	$3700-4400$	24	$12000-15500$

Table 3.1: *Bark scale - Bark bands (z) and corresponding frequency ranges*

For instance, the concept of a DNA like model which can be applied to the recognition of audio titles is presented in [NMB01]. The principle of the identification task is similar to the general fingerprint approach as described in detail in Chapter 3.2.3 and illustrated in figure 3.4: the Audio-DNA is extracted, stored and finally compared to identify audio signals. The structure of the resulting fingerprint is based on a time-line and occurring events with specified length and type. The assumption has been made that these events are discriminative enough to perform the relevant processes of identification. The following chapter will describe a similar approach with the same fundamental idea, but specialized in the use of speech signals.

3.1.3 Music vs. Speech

The comparison of music and speech in terms of recognition and identification brings up several aspects that have to be taken into account. First of all, the analysis of spoken speech is sub-classified into three methods [Vas07]. All of them are subsumed under the term "speech recognition" and often used synonymously.

- Recognition of speech
- Recognition of speaker (identification / verification)
- Recognition of spoken language

Recognition of speech

This issue deals with the capability to automatically transcribe speech signals into a digital text representation format. It is used for text creation applications. (e.g. dictation)

Recognition of speaker

The recognition of a speaker or the identification of a pre-recorded voice builds the second main approach to speech recognition. This again can be divided into two aspects, the *identification* of the speaker itself, i.e. the allocation of a present speaker to a defined group, and the *verification* that is used for systems detecting if an input signal belongs to a given authorized speaker, i.e. testing if any identification has been done properly.

Recognition of spoken language

The third interesting part is about the identification of the used language. This is applicable to spoken and written words. In case of texts this is a required feature for subsequent translation processes.

The main problem and difference between speech and music identification has its origin in the underlying model. Information that is covered by speech follows systematics which are not as simple as being applied to music or sound in general [CBMN02]. There is a sound- and word model for speech recognition as follows, but there are no defined models like this for music signals.

Phone is the smallest segmental unit of a sound.

Phoneme is the smallest structural unit that has a distinctive in terms of meaning but no distinguishing function. Phonemes are realized by phones or multiple allophones (i.e. similar speech sound variations belonging to the same phoneme).

Morpheme is called the smallest meaning-distinguishing element concerning the structure of a word.

Morphe is the analogy to phones applied on whole words.

These assumptions lead to a concept of speech that is a sequence of distinctive and abstract elements. We get our conventional understanding of speech if we add transitions of these elements and - above all - semantics to this definition.

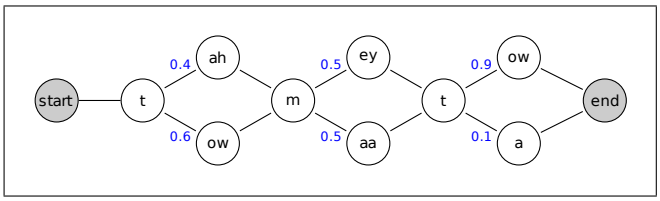

Figure 3.2: *Phonemes and transitions for the word "tomato"*

The analysis is performed by using a Hidden Markov Model on two stages. Figure 3.2 indicates a Markov chain for the phoneme analysis of the word "tomato" and its transition probabilities. Each path in this diagram shows one valid variant of pronunciation. In analogy to that the same concept can be applied to whole words, the second stage. The results are networks of both word and sentence interpretations having multiple plausibility values.

By contrast to these very basic and clear models which can only be applied to speech signals, music does not have such strict modelling primitives that can be structured and processed by automatic systems. The need for and investigations about finding a proper model is one of the most challenging tasks in the whole scientific field.

3.2 Identification

There is a variety of systems that either are subject to scientific research or actually are used in branches of audio and signal processing industry. There are two main approaches that have evolved in the last years, that are in fact able to identify pieces of music with certain accuracy reserves. These two techniques which I have dealt with are called *Watermarking* and *Fingerprinting*. Although the quantitative

and effective results of these two approaches are very similar, the principle and aim behind is totally different. While Watermarking aims at the non-audible inclusion of meta-data directly in the signal which can be extracted afterwards, fingerprinting technologies perform an analysis of the audio content itself without altering the original signal.

3.2.1 Watermarking

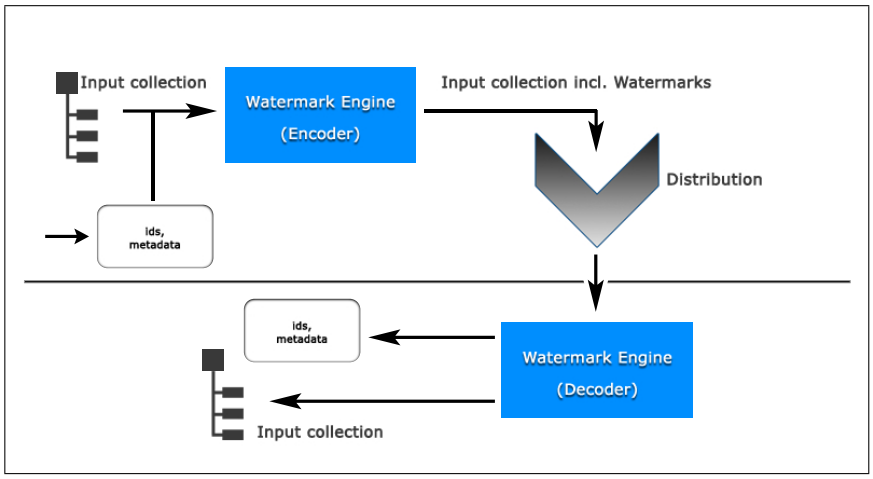

Figure 3.3: *Watermarking - encoding additional data*

The basic idea behind watermarking techniques is given by figure 3.3. Additional data (e.g. meta-data about the contents of the audio track) is included into the original file in a way that is not audible for human beings. This can be done by paying attention to the human auditory system. After the distribution the so included data can be extracted by a special watermark decoder that splits the file again into real audio information and additional meta-data. Yet, these steps alone do not constitute an advantage over the conventional situation due to the fact that the same effect is achieved with any container file formats. But, watermarking can be used also for a reliable identification after analog transmission and therewith possible distortion, bit rate decrease or other severe transformation steps. In case of a digital representation, the exceptional is that the file does not change its type. It stays an audio file and

can be played back at any time. [Cve04] [CZW08]

3.2.2 Fingerprinting

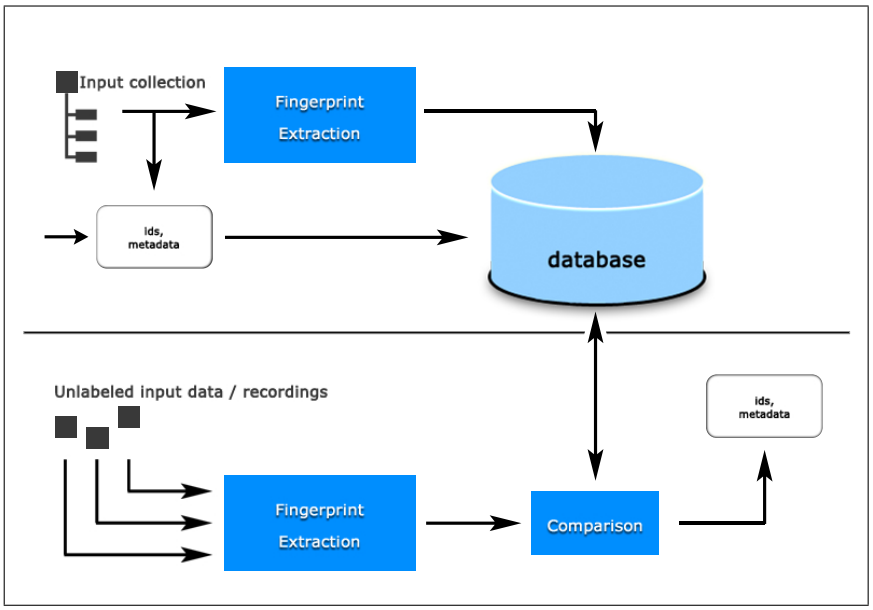

Figure 3.4: *Fingerprint-based audio identification*

Figure 3.4 provides the principle behind a classical fingerprinting approach. The upper half indicates the capturing, fingerprint extraction and database-storage steps. A title that shall be registered must be sent to the extraction block at first. The resulting fingerprint is stored maintaining an association to the corresponding meta-data in the database. The lower half shows the process which is applied to a given query signal that has to be identified. This indicates the end-users point of view. The input is processed in the same way as above. The result of the extraction block is then compared to the stored items. The meta-data of a matching item (i.e. the identified song) which is determined by measuring distance values that must conform to a given threshold are returned. The results is meta-data again associated with the query. [SBA06]

3.2.3 Fingerprint Techniques

The definition of an audio fingerprint can be seen in analogy to a human fingerprint meaning. A fingerprint that actually is an offprint of fingertips can identify human beings due to their uniqueness. Having such a copy of smallest rills on fingers let us claim that it corresponds to a certain person if we have the same fingerprint already recorded into a catalogue. The fact that such a "signature" has been recorded before any statement about the affiliation is essential - the identification is based on comparison to existing elements. [BM07] [HLH07]

In terms of an audio signal the same principle is applied. Audio fingerprints have to be extracted from the information which is contained in the piece of sound. There is no clear given feature that would be easy to just read or can be used as is. Several signal theoretical and mathematical considerations have to be made to get representations of sounds that make it possible to uniquely identify related sequences of audio frames. For this purpose, these audio features are widely transformed into bit strings or hash codes which then can be compared to each other by any given metric system. The result of this transformations is a 'small' digest of the contents of an audio signal, a kind of a summarization. The application is not limited to music contents or speech only, but covers a wide range of applications which try to associate audio information with meta-data of the corresponding content.

Baluja and Covell (cp. [BC06]) describe an audio fingerprint as an object that enables the ability to link small audio snippets with information about the content. The combination with sharing services, network based broad-casting techniques and the readiness of the consumers opens a lot of possibilities such as tagging, classification and identification. Another definition which is very suitable with regard to the applications used within the experimental part (chapter 5) and implementation (chapter 6) would define it as a 'content-based compact signature that summarizes an audio recording' [CBG03].

There are several limitations that implicate that the extraction of an audio fingerprint is not a trivial task. The fact that the similarity between different music titles can be very high or simply not discriminative enough in many cases leads to requirements which have to be fulfilled by a fingerprint extraction system. Mathematical

similarity is not inherently comparable with the similarity that is noticed by human beings. A good example here is the comparison of MP3-encoded data with raw PCM-coded audio. According to the the human auditory system, the HAS (which in fact has been responsible for the development of MP3 [IIS]) these two representations will sound very similar although the mathematical similarity has become very low due to the high data compression (about 1:11). Moreover, perceptual similarity is not defined in the same way by different individual subjects. There is no "right" way or even a model which ensures that people or systems may have the exact same idea of resemblance. In order to guarantee reasonable results in the particular field, the following properties have turned out to be desirable [Cer07], [CBG03]:

- Complexity - computational load
- Robustness
- Accuracy
- Reliability
- Scalability and extensibility
- Granularity
- Versatility
- Response time

Complexity

It is essential that an algorithm that generates summarizations of audio portions with as minimal computational effort as possible. The load of the extracting device should be applicable to thick clients (e.g. personal computers with extended hardware infrastructure - network connection, audio hardware ...) as well as to thin clients (e.g. handheld computers, integrated into electronic devices ...). The complexity covers all aspects of an identification process. This includes the extraction, comparison, transformation and all other steps which are needed to provide a result of a given recognition query.

Robustness

The term robust is derived from the Latin expression *'robustus'*, from *'robur'* which means *'hardwood'*. Robustness in terms of a proper identification attempt qualifies the independence and reliability of recognition even if the signal that has to be analysed is severely degraded in quality, or wilfully altered because of the setting in which it has been recorded. Section 3.2.4 deals with these aspects. [BPJ02]

Accuracy

The accuracy is a term specified for the quality of identification. It addresses the evaluation of the number of correctly recognized tracks, the count of incorrect identifications and false-positives analysis. The accuracy is mostly given by a percentage or mean probability that a title can be matched to its original database entry.

Reliability

According to [CBG03] reliability means that a system should be aware of the presence of a given query in the database. It should be able to give a statement about the possibility to give a result to a retrieval attempt as a matter of principle. Especially for play-list generation, real time on-the-fly song detection or library tagging, it is essential that there is a way to classify a given request as 'not identifiable' even if the false-positive rate may take an effect from that. In situations in which a database entry has not been present as yet or simply does not exist but a corresponding request is sent, the system should not give a response with a link to wrong meta-data. Instead of this a more useful reacting would be a request for the original data or using the query itself as a new database entry.

Scalability

The ability to identify and work well even if the number of database entries increases significantly is a very important requirement for audio processing applications in general. But especially the process of identification has to perform correctly also with a

large number of reference audio tracks. Respect that personal music collections often contain many thousands of music titles. To provide nearly universal recognition, systems for use by lots of people with variable interests must be highly scalable.

Granularity

The requirement that an audio snippet, which is intended to be analysed, should be as small as possible, is an essential demand. The amount of input data is an adaptable parameter for an identification system. Typically the fingerprint extraction operates with excerpts that have a length of a few seconds. In chapter 5 I will present experiment results which show that a segment size of ≈ 6 seconds can be sufficient in various aspects to extract frequency-based features that are representative enough. In [HK02], less than three seconds of audio build one fingerprint consisting of 256 subsequent sub-fingerprints that are calculated for each 11.6 milliseconds. On the other hand, the FDMF algorithm (compare section 3.2.5), an algorithm which is designed to find duplicate files concerning the contents, works on whole tracks the size of which is dependent on the length of the given query song. For use in real-time tracking or tagging applications, the amount of data or length in time which is utilized is the most important parameter at all. It has consequences on the whole extraction process and its requirements including complexity, robustness, accuracy and many more.

Versatility

Versatility and flexibility in all aspects in general is a major requirement and rule for software design. Thus it is worthwhile to design an identification application in a way that offers potential to alter or extend it. This demand is not just applicable to the number of supported audio formats or possibilities of input modes but also to database design [CBKH02]. Especially when using complex data-structures which hold information about subsequent fingerprints, versatility is essential to be fulfilled.

Response time

One main key parameter of an audio application that enables finding meta-data by query is the amount of time that is taken by the search. A simple look-up of database entries which are stored in a key-value-pair manner is not as complex, but when the database scales up and holds many thousand or millions of songs, a quick search is not a trivial task. Data-structures that contain cross-references to other fingerprints or parts of them can cause even indefinite search times that have to be determined by any threshold to stop. A search time in the order of milliseconds for over 100.000 song entries must be achieved to get wide acceptance for the distribution of commercial systems [HK02].

3.2.4 Robust Fingerprints

As mentioned above, the robustness of an algorithm concerning severe degrading of the input signal is very import to get appropriate results. Thus, a robust fingerprint has to enable the identification of audio snippets independent of many influence factors. These factors include frequency range limitations, cropping, down-sampling, compression and many more. Some of the most widespread modifications of an input signal have been tested in chapter 5.

The ideal robust fingerprint is a signature the generation of which and especially the ability of extracting relevant recognizable information works exactly like the human auditory system (HAS). Ideally, fingerprint representations of similar sounding portions of audio according to our meaning of similarity should be very similar as well. The measurement of this computational similarity is not just as simple as for humans, but with an efficient bit fingerprint (as introduced as follows), this can be done using the number of erroneous hash bits.

The same issue has been explored by [HKO01]. In this article, the following definition of a robust hash is proposed:

> *"A robust audio hash is a function that associates to every basic time-unit of audio content a short semi-unique bit-sequence that is continuous with respect to content similarity as perceived by the HAS."*

Philips Research has developed a hash extraction system that performs really well. Concerning the reliability, a false-positive rate of $3.6 * 10^{-20}$ has been achieved. Results have shown that this system is a splendid solution for identifying audio data with robust hashes [HK02]. Figure 3.5 shows the overall process of the feature extraction and bit derivation. From left to right, the signal is being framed by Hanning windows. This is a common filtering step before a Fourier Transformation to ensure that there are no "leakage" effects in the frequency domain. "Leakage" is a common source of error when applying a Fourier Transformation (i.e. especially DFT) to an input signal in the time which is not periodical or a multiple of any given ground period. This error induces non-trivial positive values for frequency bins that have no corresponding audible amplitude in the domain of time [Vas07].

The single windows are overlapping with a factor of 0.96875 which leads to a high similarity between subsequent frames. This is confirmable because the FT of audio signals with harmonic or nearly periodic amplitudes varies slowly in time. Music itself is often characterized by harmony and rhythm elements, so this assumption and the used model are consistent. On all windows the FT is applied and the resulting frequency range is divided into 33 disjoint frequency bands from $300Hz$ to $3000Hz$. This subdivision is spaced logarithmically based on the almost logarithmic spacing of the Bark scale (cp. Chapter 3.1.1) in high frequency ranges. Each band has a bandwidth of one musical tone (i.e. each band increases by the factor $2^{1/12}$ according to the scale of western music). After that, the energy of the bands is computed and delivered to the bit derivation step (highlighted gray in Figure 3.5), where the values are quantized to a bit sequence. Equation 3.3 provides the formal definition of the computation step. Depicting in words this means, that a bit becomes a "one" when the subtraction of the difference of adjacent frequency bands and the difference of neighboured frames results in a positive value. $EB(n,m)$ indicates the energy for band m of frame n.

$$H(n,m) = \begin{cases} 1 \; if \; EB(n,m) - EB(n,m+1) - (EB(n-1,m) - EB(n-1,m+1)) > 0 \\ 0 \; if \; EB(n,m) - EB(n,m+1) - (EB(n-1,m) - EB(n-1,m+1)) \leq 0 \end{cases} \tag{3.3}$$

Evaluation and measurement approaches are described in chapter 3.3.

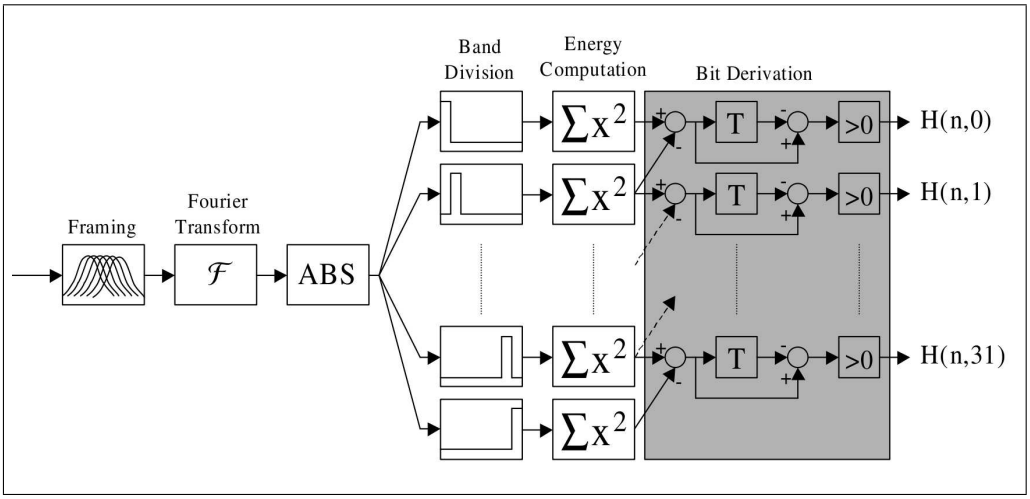

Figure 3.5: *Hash extraction scheme [HKO01]*

3.2.5 FDMF - Find Duplicate Music Files

FDMF is an acronym for 'Find Duplicate Music Files' and constitutes an open source fingerprint system provided by Kurt Rosenfeld on the MusicBrainz Websites[1]. Preliminary investigations have shown promising prospects which has led to the decision to examine the algorithm for identification quality in Chapter 5. Due to the usage for experimental identification purposes and the comparison with similarity-based approaches in the practical part I would like to describe the functionality and principle of the FDMF algorithm. It has been designed to detect equal versions of the same song title even if the meta-data is not the same and, of course, the files themselves are not totally equal. (cp. [Sin06])

Detailed operation of the algorithm

1. Decode / decompress the input file to raw audio data (PCM)

[1]URLs: http://wiki.musicbrainz.org/FDMF, http://www.musicbrainz.org, http://wiki.musicbrainz.org/

2. Apply the Fast Fourier Transformation (FFT)
3. Divide the frequency spectrum into 4 non-overlapping frequency bands (B1 - B4)
4. Calculate the energy of the bands for each 250 milliseconds chunks
5. Define a result list (FP ...fingerprint) consisting of three regions and calculate the following values:
   a) $FP[0..255] =$ The sum of the energy bands for each chunk
   b) $FP[256..511] = (B2+B3)/(B0+B1)$-ratio for each chunk
   c) $FP[512..767] = (B0+B2)/(B1+B3)$-ratio for each chunk
6. Calculate the power spectra for each array
7. Apply spline fit smoothing operation
8. Apply a one-bit median quantization on these values
9. Concatenate the bit string to get the full fingerprint out of 3 times 256 bit, a 768 bit signature.
10. Store the representation to the database (default: GDBM[2])

For the evaluation, the following steps have to be performed:

1. Extraction of the fingerprint as described
2. Comparison with database entries
3. If the results (i.e. distance values) exceed the given thresholds (one for each region), a proper identification is confirmed and printed.

The whole success of the procedure is mainly dependent on the choice of the threshold array. The three thresholds are given by $[75, 115, 85]$ by default and can be varied to support more or less fuzzy matches. The maximum and minimum values apparently are limited to 256 and 0 bit errors, the size of the 'sub-signatures'. The term sub-signatures does not refer to a temporal hierarchy of subsequent segments. The realization of these is introduced in chapter 3.3.2. The FDMF library additionally tests the input files for a conformance of the generated file digest that maps the

[2]Gnu Database Management System (http://www.gnu.org/software/gdbm/)

contents of the file at bit level to a hash representation. If these hashes are equal, the fingerprint extraction step can be skipped and a match is detected immediately because the actual file has already been treated as input.

In chapter 5 a detailed analysis and discussion of the performance and usefulness of FDMF has been performed. This has mainly be done in comparison to Rhythm Pattern, a similarity based feature extraction approach.

3.3 Retrieval and Evaluation

The final step of an identification system that has already performed the computation of the fingerprint or the extraction of a watermark is the comparison or matching process. This is the component that tests the result of an observed input signal against the pre-recorded database entries. Basically there is a subdivision of the concept into *approximate* and *exact* operation the use of which depends on the underlying identification scheme. In case of approximate or fuzzy matching, there must be a given threshold that determines at which point a result is good enough to be identified as the detected track. It seems clear that fuzzy matching has to be used because of possible distortions or altered inputs (cp. chapter 5).

3.3.1 Matching and Search

Having extracted a signature, what and why is some database key being returned? There must be a defined criterion that decides whether to return a database entry or not. The conceivably simplest solution for this problem with fingerprints of bit sequences is to count the number of bits which do not match between the query fingerprint and the database entry. This must be done for all database signatures to find the one with the minimal number of errors (compare equation 3.4).

$$E_b = \sum_{i=0}^{n} |x[i] - d[i]| = \sum_{i=0}^{n} (x[i] \textbf{ xor } d[i]) \left| \begin{array}{ll} x[i] & \text{bit at index i of input hash} \\ d[i] & \text{bit at index i of database hash} \end{array} \right. \quad (3.4)$$

The Bit error ratio (BER) in general indicates the ratio of the number of erroneous bits (E_b) to the total number of bits that are transmitted by any transport medium (i.e. the signature length). With respect to the comparison of different fingerprint bit sequences this ratio is an indicator for the overall identification success, furthermore the already mentioned response criterion.

Conventional relational database management systems (R-DBMS) store their contents in a tabular manner and have clear relations between the single elements. Search operations on such databases are based on exact matching. They perform an exhaustive search over all entries and return all elements that comply a given criteria. The simplest case here is to look for the existence of a character sequence. This approach is not usable for fuzzy-matching applications [KE04] [CBMN02].

There are solutions for this major problem described in [HKO01], [HK02], [CBMN02], [CBG03], [HKO02] and others. All of them share these main requirements for the matching and search procedure:

- Speed
- Scalability
- Efficiency

Due to the enormous amounts of persistent entries, "brute-force" search attempts will last too long as it would be acceptable to wait for. There are more intelligent approaches. One possible way is to use a look-up table (LUT) for all possible sub-fingerprints (cp. Chapter 3.3.2) that can occur (cp. Figure 3.6). These values point to the single tracks or rather the specific positions of identified sequences within the audio signal, realized by linked lists. Assuming that at least one of the signatures of an observed track is free of errors, we can find the position of the given input sequence and, as a consequence, the best match for the whole candidate data in the database by calculating the BER which must be below the given threshold. The assumption that every now and then, a sub-fingerprint of an audio piece in the database is bit-error-free has been proven. In case of no match, a Hamming distance of 1 or more can be utilized or the report of *no relevant results available* is returned.

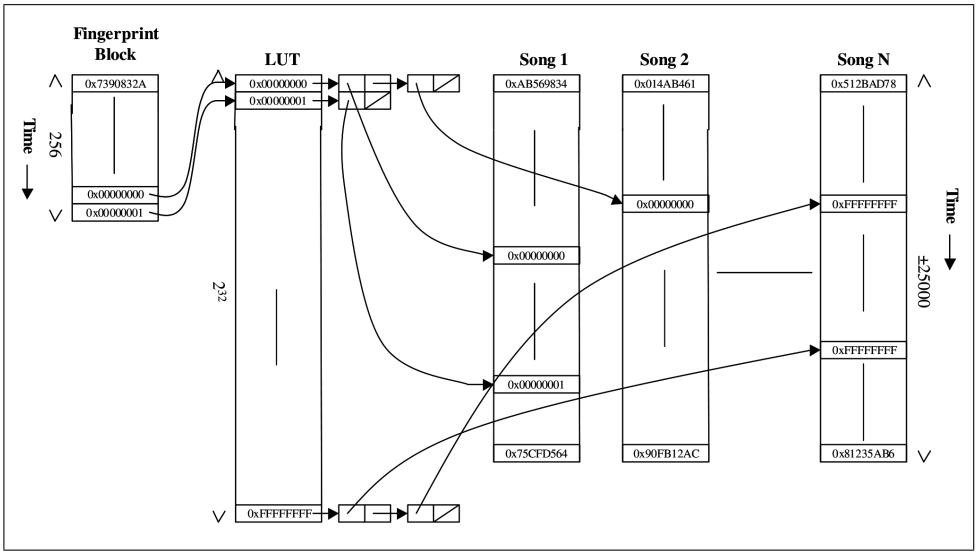

Figure 3.6: *Possible database layout for sub-fingerprint architecture using a look-up table (LUT) [HK02]*

A detailed description of the model for these assumptions and realizations is provided in the next section.

3.3.2 Hierarchical Temporal Sub-Fingerprints

As already mentioned, there is a grave limitation of the common fingerprint technology. Just reducing any extracted features out of an audio track and converting them into a bit representation results in signatures containing absolutely no information about the temporal occurrence and sequence order of acoustic events. This issue is assimilable to the Fourier Transformation in general very well. The only way to maintain some general time information is to subdivide the input signal into multiple parts, analysing them separately and merging the results again in a way that preserves the temporal data. Figure 3.7 shows the temporal evolution of Cosinus functions with varying frequency every 55 time units (i.e. samples). Haitsma and Kalker introduce a fingerprint system in [HKO01], [HK02] that takes care of exactly this problem. The extraction of the whole robust hash which identifies a track consists of a variable number of so-called sub-fingerprint blocks which again

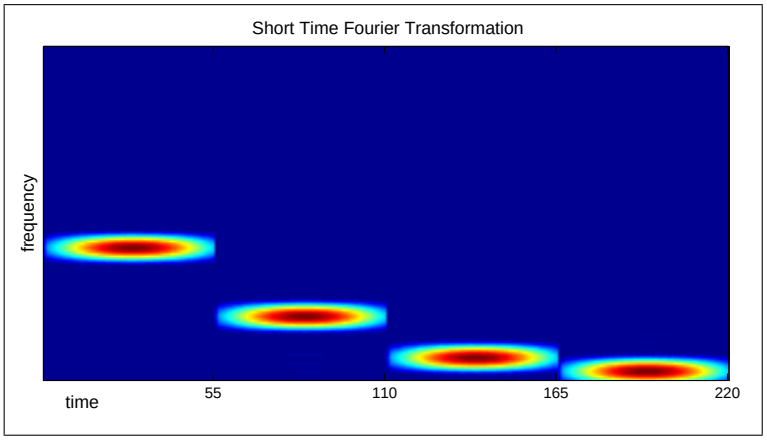

Figure 3.7: *Short Time Fourier Example - Frequencies for temporal intervals*

are divided into single sub-fingerprints (SFP). These SFPs have a encoded length of 32 bits which correspond to a temporal length of 11.6 milliseconds. 256 of these SFPs build one SFP block ($11.6 * 256 \approx 3$ seconds).

Considering the database layout in chapter 3.3.1, illustrated in Figure 3.6, this approach is useful for both finding a certain position within the piece of audio of a database entry while analysing and identifying audio signals taking just about 3 seconds of data.

Chapter 4

Audio Similarity Feature Sets

The second main approach which I have dealt with in this thesis treats the possibility to use similarity based feature sets not just as a method to get distances in terms of musical similarity as it can be used for building classes of similar music like genres or retrieving acoustical neighbours, but as a useful technique which originates in the capability of identifying music. Further on I will give an overview about the features themselves as well as measuring, retrieving and ranking the results. As an exemplary implementation, I investigated the 'Rhythm Patterns' feature set for its practicability in audio identification compared to a dedicated audio identification algorithm. The basic functionality is described due to later usage in the empirical part.

4.1 Audio Feature Extraction

Data about or describing other data in the common sense is called meta data. These can be additional information about contents, hardware related, formats and other useful and processable information. In case of an MP3-audio file, ID3-tags hold content related properties that have been added manually (annotations). Being able to extract or generate meta-data automatically induces the term *feature* which in fact is a characteristic property of the referred data itself.

A feature can contain a variety of information that is inherently included in the original signal. Common properties like tempo, length and others are also possible

extraction results as well as features that can not be comprehended by humans as easily, such as spectral flux, roll-off, zero crossing rate and many more. [TEC02]

Musical features in general are measures for specific characteristic interests of parts of audio signals. For classification or identification purposes the assumption can be made that proper feature combinations and weights build a distinctive description which can be used to distinguish music tracks from other, different sounding titles. Additionally the grade of difference - the distance - constitutes a measure for similarity. [FGDW06]

The challenge for many applications then is to find, define, extract, combine and eventually transform single values into powerful music descriptions that are used for the tasks mentioned.

4.1.1 Common Musical Features

Semantic level and classification

Meta-data or additional semantic information can be classified using different approaches. A subdivision into *high level* and *low level* data is often used to delimit annotations which are done by hand from technically extractable information. Referring to this differentiation, high level features have to be used for systems that should lead to any advantage. According to the semantic level of description of the relevant objects, the features do not have to be present in a numerical form but in any data type that enables the storage of the relevant interests. [AHH+03] [HBW+08]

[NL99] proposes a classification scheme that has been developed under the research activities for MPEG-7 descriptions. The examples have been chosen according to audio related scenarios.

Medium-based

the description of the medium / the form in which the data is expressed (e.g. sampling rate, word length of quantization)

Physical
: computational features that can be extracted out of the raw audio material (e.g. loudness level, power in frequency domain)

Perceptual
: grouped and derived from the physical aspect; characteristic features that are present in a form which corresponds to the human perception or are in fact perceivable (e.g. timbre)

Transcription
: the semantically enhanced data that has been created out of the contents or is a reconstruction of the object (e.g. lyrics)

Architectural
: defines the syntactic structure that is needed for building semantic descriptions on lower levels

Annotative
: additional annotations of human beings which would not be extractable automatically

The abstraction level gains from top to bottom. *Medium-based* data is located inside the object itself where *annotative* has to be generated arbitrarily by humans.

Musical features that are intended to be used for similarity measurement or identification purposes with respect to the domain of music information retrieval are located on the physical and perceptual layer. The extraction of them is chiefly done in the time domain or frequency domain. The appropriate transformation that converts the time based signal into frequency information respectively its inverse procedure is performed by the Fourier Transformation (DFT for discrete signals). Possible and useful features are outlined in the following sections.

The frequency domain

Many properties and features of music can only be obtained by analysing the frequency domain. Although they have their main characteristics that are remarkable for us humans in the time domain - music is considered to be amplitude variations

over time - or as a representation of changes of the amplitude of different frequencies, a conversion into the frequency domain must be done which gives a deeper insight into amplitude variations on different frequency bands. This enables the extraction of features that would not be extractable otherwise. The formal definition is illustrated in Equations 3.1, 3.2 in Chapter 3.1.1. [BK07]

Commonly used techniques perform parts of and extensions of the following fundamental steps:

- Subdivision of the audio signal into frames of a specified window size *wS*
- Windowing using Hanning windows, Hamming windows or another function that lowers the amplitude towards the borders. This reduces the artifacts on the frequency side such as leakage effects.
- Transformation from time to frequency domain using the discrete Fourier Transformation (DFT), discrete Cosinus Transformation (DCT), Wavelet Transformation, ...
- Subdivision into multiple bands (frequency ranges)
- Calculation of features from the spectral data
- (Summarization)

By these and other transformations [GDPW04], features can be extracted that include characteristical descriptions about rhythm, tempo and many other aspects. A well-known feature is the BPM value - beats per minute. It specifies the overall tempo which is mostly given by percussion or bass instruments. By contrast to the *naive* tempo, it calculates the feature in a more intelligent way than just considering the re-occurrence and periodicity of the highest peaks that can be found. Beat histograms illustrate the deviation of intensities of beat-supposed elements over the whole frequency range. A high value for one bar then shows the tempo and the responsible frequency lots.

The problem for all rhythm-based features is the fact that there is no exact association of rhythm or periodicity to any fixed classification or musical division which would consequently lead to a recognition of the genre. Although different musical styles are often characterized by the tempo and periodicity, there are too many

styles available to clearly associate simple rhythm descriptors with a genre. An indisputable connection of rhythm as is with the identity of an audio title is not possible. As mentioned before, there must be a proper combination of multiple sets of features to model the human identification ability.

As an example, in [TEC02], genre classification is performed utilizing a characterization of musical texture and timbre as well as instrumentation. The extraction process produces a 9-dimensional feature vector containing values that are calculated using the STFT / FFT in the frequency domain. Time domain based features are also part of this categorization process. Therefore the input signal is subdivided into texture windows with a length of 1 second that contain 20-milliseconds analysis windows. Mean values and standard deviations are based on the occurrence and comparison of these analysis windows.

Mean and standard deviation of centroid
: are values which indicate the spectral brightness of an audio signal

Mean and standard deviation of roll-off
: constitute measures for characterizing the spectral shapes

Mean and standard deviation of flux
: represent descriptions of change in the spectral domain

Mean and standard deviation of zero crossings
: indicate the dynamic behaviour of a signal by counting the number of crossings of the "zero line" (i.e. change of the signum function) in the time domain

Low energy
: describes the ratio of the count of analysis windows that have less energy than the average over all in a *texture window* by all 40.

These combined features are used to classify musical genres. The quality of this process has been tested using confusion matrices as well as 10-fold cross-validation. For a complete description of the calculation and evaluation, refer to [TEC02].

4.1.2 Rhythm Patterns

Basically a Rhythm Pattern is a representation of sinusoidal audio contents using a relation of critical bands to the periodicity on critical frequencies according to the human perception. The full range of critical bands leads to a matrix dimension of 24 frequency bands $\times$ 60 different modulation frequencies for amplitude modulation $= 1440$ values. The so included information covers the common sense of rhythmic elements as well as frequency based regularities.

Due to the usage in the experimental part of this thesis, Rhythm Patterns (RP) are introduced in detail.

The extraction process is illustrated in Figure 4.1 and described as follows [Lid06] [RPM02]:

First of all, input data that should be analysed has to be preprocessed beforehand. Depending on the underlying application scenario different procession steps can be performed. Normally this process includes stereo to mono conversion, initial segmentation, skipping of lead in or lead out (etc.). [KQG04]

The Rhythm Patterns extraction processes audio data in chunks of ≈ 6 seconds of length which correspond to 2^{18} samples at a sampling frequency of $44.1kHz$ (for lower sampling frequency, the number of samples is halved linearly to maintain the 6-seconds segment size). Each of the segments is transformed using the Short Time Fourier Transformation (STFT). The FT-windows are Hanning filtered with an overlap of 0.5 at a length of 23 milliseconds. The frequency range of the signal is then divided with respect to the Bark scale into 24 Bark bands. This is done due to psycho-acoustical issues i.e. the loudness perception of human beings.

After spectral masking, conversing to the dB-scale, loudness equalization and the transformation to Phon[1], the critical bands are now analysed separately to get modulation frequency parts. For each band the FFT provides a time-invariant representation which in fact contains the rhythm information (like pitch, tempo, BPM ...).

[1] Phon is a unit for the loudness level with respect to the human perception introduced by Stanley Smith Stevens (1906 – 1973)

Final filtering and smoothing operations lead to the Rhythm Pattern representation as described. [SWS07]

The resulting matrix can be used for similarity measurement. There is the assumption that a small distance value between two Rhythm Patterns consequently implicates high similarity concerning the human perception. Definitions for different similarity ideas are provided by Chapter 4.2.

4.2 Audio Similarity

Similarity in the common sense refers to an assessment of relations of descriptions of different elements in a subjective way. What a human being calls *similar* to something does in general not conform to opinions of others. This statement seems to be very trivial and self-evident, nevertheless it is a non-trivial challenge to define a model that enables a formal measurement and evaluation of the term similar. [AP02]

Grouping elements having a relative high similarity leads to kinds of clusters which represent one mean description. The level of cohesion is then given by the deviation between these objects. The main problem here is not to find a criteria which is used for the clustering procedure, but the fact that there are a lot of possible influence factors that exclude each others. Trying to maintain a kind of rhythm information will possibly neglect pitch characteristics or dynamics.

In terms of audio tracks, similarity can be defined using a variety of features that are extracted. A proper combination of them is the main goal for getting a useful feature vector that really models our sense of musical similarity. The most widespread meaning of similarity between audio tracks is the genre. The classification and usage of this semantic descriptor is outlined in Chapter 4.2.5.

4.2.1 Spectral Similarity

According to [LS01], there is a basic way to compare distinct audio titles while only considering the audio contents. The division of the signal into multiple frames with

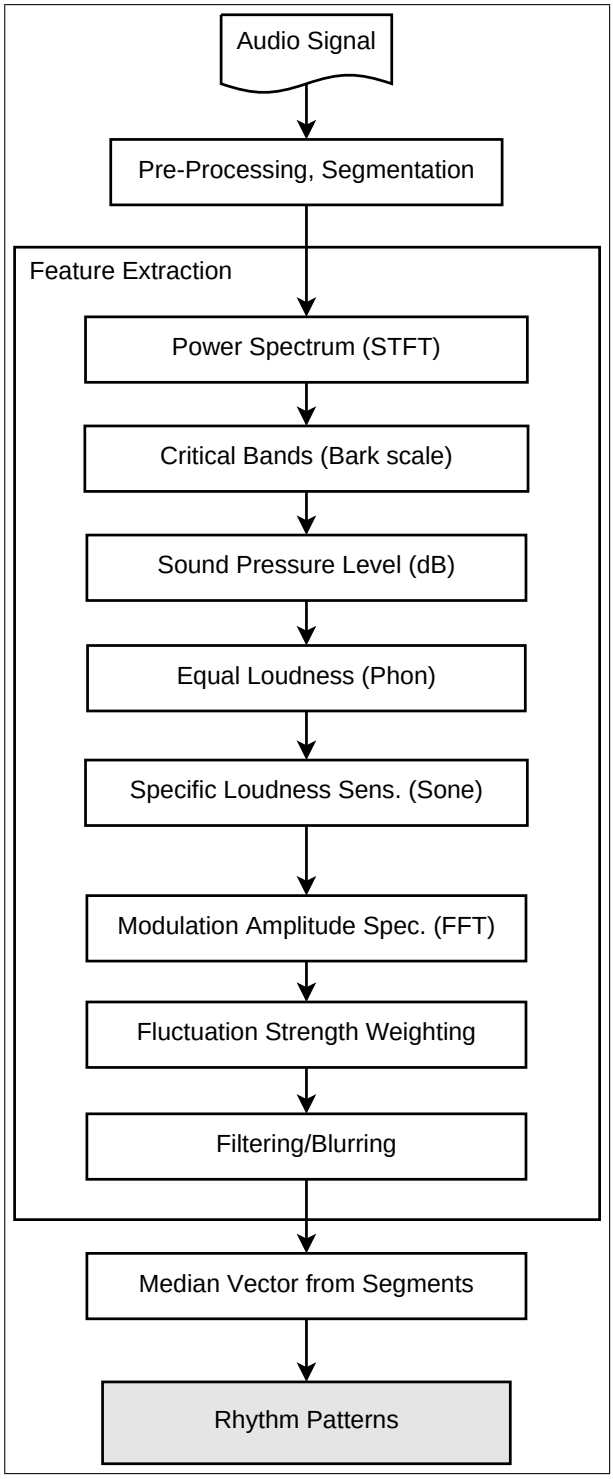

Figure 4.1: *Extraction of a Rhythm Pattern [Lid06]*

specific size, the conversion into a spectral representation that covers the sinusoidal parts as well as the subsequent clustering of the frames results in an audio signature based on the spectral domain. Among others, this approach has become one of the most popular ideas behind audio content similarity measurement in the whole scientific field. (cp. [FGDW06])

4.2.2 Rhythmic Similarity

Rhythm can be defined by a general structuring of tones, sounds or even movements into sections or parts which are concerning the contents that are even, similar or just re-occur periodically in an equal or similar way.

This term contains several aspects that have to be taken into account. Rhythm does not just cover the simple temporal issues (i.e. speed of periodical events like bass drums etc.) but also the kind of beat as there are multiple musical bars and the variations of them [GDPW04] [FGDW06] [TEC02]. Just being similar in terms of the rhythm does not consequently implicate a subjective, overall similarity. The tempo or rhythm alone therefore are no clear nor general descriptors for music similarity.

4.2.3 Semantic Similarity

Another point of view of similarity also considers the semantics of a piece of music that does not originate from the contents - the context. Although this has absolutely no relevance concerning the audio content itself and the extraction of content-based audio information, the semantic information and context is an often underestimated criteria for similarity retrieval and identification systems. It is obvious that existing semantic information cannot be included into real content-based approaches but in fact there are semantic features which would be extractable in an automatic way. Intelligent system would benefit from that. In [Lei04, pp. 69] users show a totally different understanding for declaring two songs as *similar*. For example, the used speech in some of the example tracks is a criteria which differentiates different description groups or genres. There are approaches which deal with the detection of

languages (cp. [Vas07]). A possible combination of these extraction ideas would lead to an advantage at least in this special scenario.

4.2.4 Distance Metrics and Representations

In the mathematical sense, a distance is the length of a direct connection between two points in any dimensional space. Hence, it is a function that has two input parameters and assigns one numerical value to the pair of input data that indicates the distance. Equation 4.1 shows the formal definition for the L2- or Euclidean distance which is widely used in music information retrieval. It has been used for the experimental part of this thesis (cp. chapter 5).

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + \cdots + (x_n - y_n)^2} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{4.1}$$

The input vectors are given by:

$$X = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^n \quad \text{and} \quad Y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n \tag{4.2}$$

Picturing illustratively, the common spatial distance in the 3-dimensional space ($\mathbb{R}^3$) is given by the choice of a 3-dimensional vector for the elements ($n = 3$) that have to be considered. A flat Cartesian coordinate system with 2 dimensions uses a 2-element vector. As Equation 4.2 illustrates, the generalization (the L2 distance) is able to compare n dimensions. It is an appropriate metric for pairwise comparison of feature vector elements with the special case of equal weights for all features and - of course - the same number of elements on both sides, the query and the database entries.

Self organizing maps (SOM)

A popular technique for managing high dimensional data is the Self Organizing Map (SOM). It is a type of an artificial neural network that reduces the dimensionality of high-dimensional vectors down to (usually) 2 coordinates on a flat grid. This kind of representation has a lot of benefits as it is much more comprehensible for human beings than high-dimensional vectors. It can be post-processed in various ways to produce other useful representations (cp. [VHAP00]). In case of audio data and extracted feature vectors, it is a useful technique to map the underlying audio tracks which are represented by the neurons onto a map which illustrates the similarity of neighboured titles.

Therefore, a sequential training process must be performed which is responsible for the topology and structure of the map. Each iteration rearranges the neurons and quasi unfolds the grid. There are weight vectors (one per unit / neuron), the prototype vector, the feature vectors are associated with. They have to be determined and influence the entire arrangement as they are connected to the adjacent neurons via neighbourhood relations which are trained. The single steps are outlined as follows: [VHAP00]

- Choice of trainings vector x randomly
- Computation of distance between x and all weight vectors of the SOM using any distance measure (common Euclidean)
- Determination of the best matching unit (BMU) by using the smallest distance to all others
- Update of the weight vectors so that the BMU gets a closer position to x than it already has. The neighboured elements are treated similar to maintain a relatively equal but stretched topology (cp. Figure 4.2)

Steps 2-4 are carried out iteratively.

In general, the whole training is performed in two main stages, the approximation of a proper order as well as a fine-tuning iteration sequence that determines small variations leading to better results.

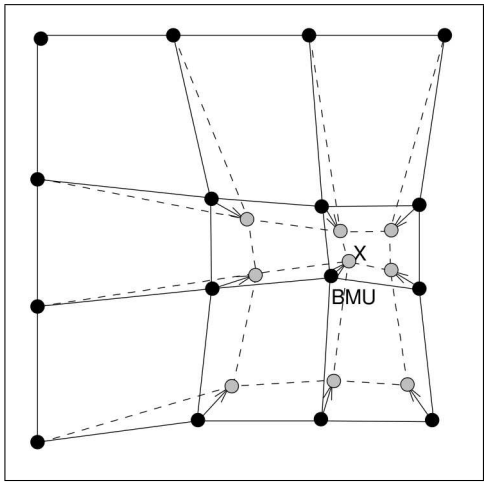

Figure 4.2: *Update of the best matching unit (BMU) and adjacent neurons according to input data x [VHAP00]*

4.2.5 Genre Classification

> *Musical genres are categorical descriptions that are used to characterize music in music stores, radio stations and now on the Internet* [TEC02].

This definition of the term "genre" just describes the common meaning of the most widely used classification scheme in music at all. Although the understanding is somewhat subjective and cannot be generalized, music genre classification is a large scientific sub-field of music information retrieval. As described in chapter 4.2, the choice of the proper features and the concentration on the relevant parts of an audio signal is not trivial. Therefore no exact statements can be made. Similar descriptions of similar sounding titles are a general requirement which must be fulfilled. [Kos02] [KQG04]

In [TEC02] it was pointed out that a combination of various musical views and their corresponding features lead to quite satisfying results of a proper classification that models our common meaning of similarity and the genre. Musical surface features (described in chapter 4.1.1) together with rhythm features build the descriptor for the introduced feature extraction (cp. [TEC02]). There is also an implementation of a real speech-music discrimination and separation. A hierarchical arrangement of

musical styles as well as the speech have been used and treated differently according to the recognized type of the signal on the coarse level. [FGDW06] [RPM02]

There are comparative studies about the ability of classifying musical styles. Humans are able to identify a genre - respectively assign music tracks to pre-defined genre terms - astonishingly well. This does not just include real active listeners or experts in the musical field, but also music interested people that have no specialised knowledge [Kos02]. On the other hand - referring to the definition and aim of genre classification - it is the only apparent, even logical assumption as building models always relies on the human understanding.[MB03]

4.3 Considerations for Evaluation

Retrieval systems in general have a clear structure that is visible to the "end-user". Besides the whole extraction process, the pre-procession, post-procession and others as illustrated in chapter 4.1.1 and 4.2, the retrieval engine builds or receives a query and sends it to the database. The comparison engine again can modify or transform the input, but in general computes distances between feature values and produces a result set in a specified format. In case of identification or similarity measurement, a list of candidates with ratings is returned. While it is the purpose of retrieval-by-similarity to return a ranked list of similar results, an identification algorithm is supposed to deliver the one identified song - a single result.

In many cases, the query is not given in a machine-processable form as is. Hence, the query engine must generate a request that can be dealt with out of the content information. Two main examples which alter the overall view on the identification procedure are given as follows. These aspects just concentrate on the retrieval, the extraction of features from all music tracks in the database must be performed anyway. [Foo99] [MG06]

Query by humming

The input data is an audio signal (commonly recorded by a microphone) which has been produced independently of the searched original file. There is no

transformation and therefore maintenance of signal parts as the data is produced completely differently. The retrieval of similar sounding descriptions must be done by very robust and intelligent solutions when having the focus on the extraction of features. MIDI-like sequences of musical tones and their temporal offsets as well as pitch differences can be considered too. [DG06]

Query by example

This term is nothing else than another view on the content based audio retrieval as described. A single musical piece or an excerpt is used as the input for the query. No transformation has to be made of the signal and the feature extraction can be applied directly. The input has to be considered, processed, features are extracted which are combined to build representations that can be compared numerically.

4.3.1 Evaluation Measures

The responses of a retrieval system must be able to be measured and interpreted in order to make statements about the resulting quality. The following evaluation measures are based on the false-positive analysis and are used for the information retrieval domain as well as clustering and classification. The quality or relevance of the individual resulting elements is not included by this evaluations methods. Due to the limitation of certain systems that a result can only be true or false, these measures are used for Boolean retrieval. Figure 4.3 illustrates the relations between the different sets using Venn diagrams. R indicates the set of relevant elements, A stands for the answer set. The intersection and union sets are used for the evaluation.

Per definition, identification means that a specific element (i.e. audio track) is found. Due to the assumption that this must be the database entry with minimum distance or maximum similarity, a ranked answer set can be provided where rank #1 should be the title in demand. Precision, recall and fallout are computed to rate whole answer sets as well as the single results.

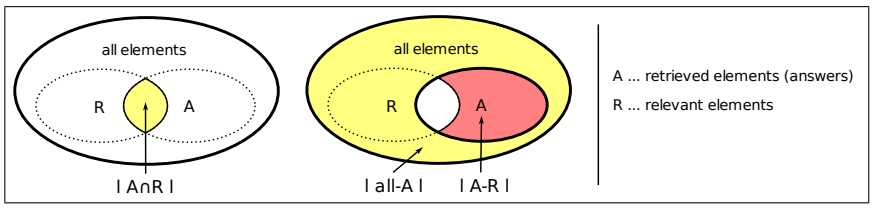

Figure 4.3: *Result elements and intersections*

Precision

The *precision*-value is given by the ratio of true-positives by all retrieved documents. It corresponds to the statistical terms of relevance or positive predictive value. The formal definition is given by Equation 4.3. The resulting p-value lies between 0 and 1 where 1 implies that all retrieved elements are relevant. For identification purposes commonly a ranked answer set is provided. The number of results can be increased incrementally as long as the precision value for the last retrieved element tops a pre-defined distance threshold. [DG06], [Vas07]

$$\text{precision} = \frac{|\{\text{retrieved elements}\} \cap \{\text{relevant elements}\}|}{|\{\text{retrieved elements}\}|} \tag{4.3}$$

Recall

Another characteristic value for evaluation of retrieved elements is the true positive rate (TPR) or *recall*. According to the statistics this is called the sensitivity of a process or operation which is expressed by the relation of relevant retrieved elements to all relevant elements in the database (cp. Equation 4.4). A value of 1 corresponds to the fact that all relevant elements have been found by the system. An identification attempt (i.e. recognizing one specific track) performs perfectly at a recall of 1.

$$\text{recall} = \frac{|\{\text{retrieved elements}\} \cap \{\text{relevant elements}\}|}{|\{\text{relevant elements}\}|} \tag{4.4}$$

Fallout

The semantic inversion of the recall is given by the *fallout*-value or false positive rate (FPR): the ratio of irrelevant found elements to all irrelevant elements (cp. Equation 4.5). It is mainly used for evaluating the negative performance.

$$\text{fallout} = \frac{|\{\text{retrieved elements}\} \setminus \{\text{relevant elements}\}|}{|\{\text{all elements}\} \setminus \{\text{relevant elements}\}|} \tag{4.5}$$

F-Measure

A much more important characteristic value is the F-measure, which is a weighted harmonic mean combination of recall and precision. The weights for the two influence values are controlled by one parameter β. The bigger it is, the more recall is respected, a value of 0 lets the f-measure correspond to the precision-value.

$$F_\beta = \frac{(1+\beta^2) * (\text{precision} * \text{recall})}{(\beta^2 * \text{precision} + \text{recall})} \tag{4.6}$$

4.3.2 Database and Ranked Retrieval

The pivotal part of the whole extraction and comparison procedure is the ordered representation of different database entries which principally are worth considering. Multimedia databases in general have a fuzzy retrieval nature which means that no exact results can be given compared to the conventional way of relational databases. This is an important assumption that has to be made. Content-based queries always rely on the similarity or distance metrics that determine whether to return a specific element as relevant or not. To ensure that a pre-selection of potentially unimportant candidates is made before any result is found, there are several strategies that can be applied. These cover *Clustering*, *Query relaxation* and *Reformulation* which are described below. By contrast to conventional multimedia databases, the retrieval respectively the identification of audio material follows the strict requirements of a real content-based system. Therefore no additional information is provided. There is absolutely no combination of traditional query elements with the content-related

features. This makes things much more complicated but ensures that semantic data has no influence on the quality of feature extractions. Possible conversions from features to semantic information, i.e. the abstraction of numerical values to a coherent and usable semantic statement can improve the human ability to retrieve the right elements since it is the human assessment that rates a retrieval system as successful or not.

Clustering

A really important approach not just for identifying music but for information retrieval and classification in general is the clustering of elements that are featured by any similar properties of their own. The choice of the relevant extracted features is no fixed criteria and can be varied. Using multiple feature descriptions and defined classes this leads to classifications on various levels and abstraction grades.

In many cases, the clustering is based on semantic information. This is done due to the similarity understanding of human beings. A cluster of elements given only by the same extracted feature value does not really give insight for a subjective evaluation. So there must be a relation and conversion from numerical values to semantic interpretation considering the content-based analysis of music material. The best example here is the classification of different music styles using genres. There is a roughly common understanding - of course not totally consistent over multiple human beings - about what a musical style is about and what is describes. Otherwise is would not be possible to define style classes at all. This kind of information can be extracted and converted into genres which in fact is a big research area in the MIR (music information retrieval) field. [Kos02] [TEC02]

Concerning database search strategies, clusters can be used for prematurely selecting a potentially useful group of elements. This leads to an early assessment of the number of candidates and therefore feedback at an earlier point of time. Remember that the time for an identification attempt is an essential criteria for quality in retrieval systems. Additionally, an association of an element with a pre-defined class may improve the whole search procedure by enabling the system to use different metrics, comparison approaches and even algorithms. This again can accelerate

the retrieval and leads to better performance. To extend the clustering procedure further, relaxation and reformulation can be applied. [MG06]

Query relaxation and reformulation

There are several techniques that are able to alter given queries to fulfil database-related assumptions or improve the whole search process. Query relaxation produces another query by lowering the exact matching assumption to a more fuzzy comparison. Looking at the comparison techniques, this in general means that a certain threshold is chosen together with a discrimination function in a way that actually more results can be found. Hence, the whole query is made *"fuzzy"* dependent on the underlying content and document format.

Having a look at the evaluation of Boolean retrieval techniques in chapter 4.3.1, it comes out that a variation of a query in order to retrieve more or less results consequently implies that the quality measures are influenced. "Relaxing" the query too much will induce that too much non-relevant results are included into the answer set. By reverse, applying too humble relaxations can cause that no relevant elements can be found and therefore no identification can be made. [MG06] [DG06]

The approach of "reformulation" considers the query not to be alterable. Hence there must be a total reformulation to get reasonable, more or generally results. [MG06] Having a look at audio retrieval systems, a reformulation induces a total exchange of used features or a capture of another part of the regarded signal. Considering audio streams, the whole identification system could wait for some seconds, capture the input data at another position and start the whole analysis procedure again.

The important task of a retrieval system and therewith the whole database-connected software is the confirmation that results are provided which fulfil the defined criteria. Depending on the input data, the content and the database strategies, there must be a way to enable to make an assessment to permanently improve music information applications.

Chapter 5

Experimental Results

In the following chapter I will describe the practical and empirical part of my thesis. I want to introduce a series of test cases which have been performed during the quantitative and qualitative analysis of the algorithms. The aim is to figure out in what sense a variation of the observed amount of data or a signal modification results in better or worse accuracy values. The cases are built up in constructing manner and explain the basic functionality as well as the effects for identification purposes. Identification in this case means that one specific audio data stream is analysed and due to various comparison techniques considered to be equal to an existing track in the database. The feature extraction methods are described in Chapter 3 and Chapter 4. This is done both for the RP [1] and the FDMF [2] algorithm. Furthermore I will provide an interpretation of the results and give some explanations about the resulting consequences and provide general conclusions.

5.1 Preliminary Considerations

5.1.1 Audio Stream Models

One special goal of the analysis is to figure out, in what sense the content of an audio stream can be identified correctly using either a Rhythm Pattern or the FDMF algorithm. Having Matlab as the computing tool, it is necessary and essential to

[1] RP ... Rhythm Patterns
[2] FDMF ... Find Duplicate Music Files

build a simple model which conforms to the situation we have when capturing real streamed data that can be expressed by Matlab constructs. For this purposes, the test set (compare Table 5.2) is regarded as streamed audio.

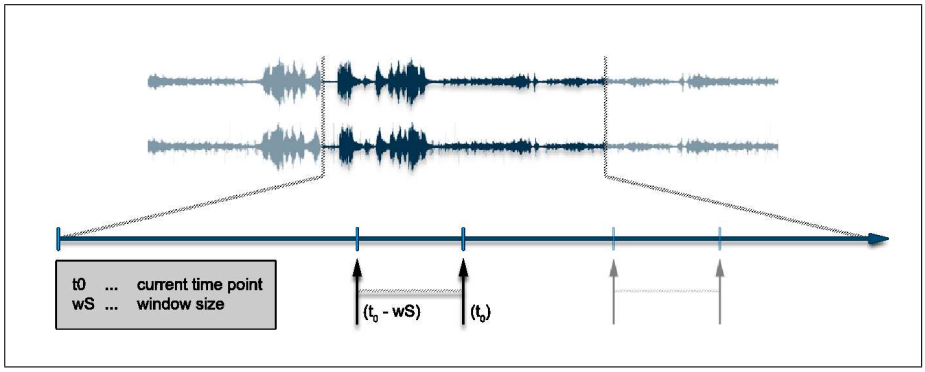

Figure 5.1: *Scheme for analysing streams*

The current point in time t_0 is chosen randomly after a minimum of wS samples, the considered window whose included data is used for the analysis steps has its window size wS. As Rhythm Patterns are extracted for segments of fixed size sS (2^{18} samples), the window size of the stream must be a multiple of this segment size. ($wS = x * 2^{18}$). The resulting extract is given by the interval $[t_0 - wS; t_0]$. This kind of consideration, as the left bound is stated by a subtraction of the right bound, conforms to the assumed real world scenario in which the user normally wants to get the information about an audio signal which is currently playing at a specific point in time. For special cases in which there is not enough data available yet at t_0 - in other words $t_0 < wS$, the system will have to wait until wS samples are buffered and ready to be processed. This approach is similar to the audio capture implementation of the JAVA application (compare Chapter 6). Again - at a specific point in time t_0, data has already been buffered and can be retrieved as a byte array of raw audio data until point t_0 with buffer size bS. Figure 5.1 shows illustrates a simple view of this assumed model.

5.1.2 Compressed Audio Formats

When using compressed audio formats as a transmitting format the incoming data has to be decoded beforehand. Although most of the codecs are not loss-less there is still enough data to perform tests on it. The portions of audio material have to be decoded using appropriate libraries (as it has been done in Java for the implementation - Chapter 6) or external applications (so done in Matlab for the experimental part). Streaming directly to raw PCM format solves this problem very smartly, because the local buffered data is already present in the preferred form and can be used without further procession steps.

5.2 Test Case Introduction

All experiments, which have been performed and are described in this chapter, have been written in Matlab. The M-Code-Files are available in the code archive, which is supplied on the complementary web page for this thesis[3]. Each of the test cases is scripted in standalone manner for later comprehension and reproducibility. The resulting figures based on the values produced by the Matlab scripts can be provided on demand.

5.2.1 Modifications

The main goal and a very important parameter that has to be minimized for an audio identification system which is based on the contents is the amount of data that has to be taken into account. Normally the input signal in terms of time should have a length of some seconds. This factor is investigated in detail in chapter 5.3. The second main aim, besides having deeper interest in the amount of data which is regarded for the identification experiments and in how far the segment size has an impact on the resulting precision, is the influence of signal modifications on the results. Several of these audio signal variations have been tested to ensure reliability

[3] http://www.ifs.tuwien.ac.at/mir/audiofingerprinting.html

despite severe data degrading. These modifications cover pitch cue variations, frequency filters and changes in the dynamic behaviour [CBMN02]. Table 5.1 gives an overview. In Section 5.3.1 more information as well as the experimental results can be found.

frequency	frequency filter (analog equalizer [a]) band pass (GSM [b])
tempo	pitch cue $\pm 1\%$ pitch cue $\pm 3\%$ pitch cue $\pm 8\%$
dynamics	compressor

Table 5.1: *Signal modifications*

[a] Analog equalizers are widely spread when using analog mixing devices

[b] The GSM scenario simulates a limited frequency band transmission with a butter-worth band pass filter: $[\approx 300Hz; \approx 3kHz]$

5.2.2 Test Set - Ground Truth

For all experiments the same standardized test set is used (so-called ground truth). This is recommended because of comparability and uniformity within the scientific community, which wants to participate and benefit from the results. The set used is the one that has been offered for the **ISMIR 2004 Audio Description Contest**[4] as *Development Tracks 1 & Development Tracks 2* for the genre classification task. It contains 729 music files, which are available in the specified format:

For all extraction processes and tests, the audio data (originally provided in MP3-encoded format - see Table 5.2) has been decoded to *Uncompressed 16-bit PCM audio* using MPlayer for Ubuntu Linux [5]. The uncompressed total size of the ground truth amounts 29.0 GB (2.6 GB encoded), in terms of time about 49 hours.

In the following sections, the term *query* is called the observed input data which has to be analysed and tested against the database. *Prototypes* or *references* refer to the already recorded database entries.

[4] http://ismir2004.ismir.net/genre_contest/index.htm

[5] MPlayer 1.0rc2-4.2.3 (C) 2000-2007 MPlayer Team

Codec	MPEG 1 Audio, Layer 3
Sample rate	44.100 Hz
Bit rate	128 kbps
Channels	Stereo
Number of tracks	729
Minimum length	842.112 samples (≈ 19 secs)
Maximum length	68.353.920 samples (≈ 1.549 secs)
Average length	10.678.944 samples (≈ 242 secs)
Median length	9.536.256 samples (≈ 216 secs)
Genres	classical electronic jazz/blues metal/punk rock/pop world

Table 5.2: *Source format of ground truth*

5.2.3 Measurement of Performance

The performance of the introduced test cases is measured by relative amounts of correctly identified audio track instances using each of the 729 songs as query title for identification. The results are given by their corresponding percentages (accuracy). For the experiments, the single audio titles are selected and treated as queries. For all files in the database (references), the distances to the observed signals are calculated and stored in an array. Sorting this array provides the best match on index #1. The distances are measured by 'Euclidean metric' as follows:

$$\sqrt{\sum_{i=1}^{n}(x_i - d_i)^2} \quad \left| \begin{array}{l} x_i \ldots \text{prototype values} \\ d_i \ldots \text{database values} \end{array} \right. \tag{5.1}$$

This way of getting distances between high dimensional feature vectors is used for the comparison of decimals providing an option that takes all single values (of the vector) into account and respects their specific distances - it is not a simple summarization. In cases of the utilization of a bit representation (compare Section

5.3.6 and Figure 5.4), the distances are expressed via simple bit errors which in fact is a term for the XOR operation or just a special case for the Euclidean metrics. Bit representations - in our case fingerprints - are used for reducing the overall amount of information available. The numeric values are transformed (e.g. Median Quantization, compare Section 5.3.6) into a sequence of bits which cover the same information set but in a form which has a much lower demand of both system memory while processing the fingerprint as well as hard disk memory for persistent storage in a database. There is also an improvement of the overall procedure concerning the response time, the search speed and the use of computational resources which is quite important for real-time applications or just time-effective systems that should be able to be executed on personal computers.

5.3 Results and Experimental Performance

The test cases are divided into multiple classes. First of all the cases will introduce a deeper understanding of in how far a change of the segment size results in a better or worse precision of an identification process. The following experiments show how acceptable performance and reliability can be achieved.

5.3.1 Detailed Realization

Each experimental step and test case is described in detail and the results are presented both in a graphical and tabular form. The experiments cover tests which bring up several aspects about the *Rhythm Patterns*, the *FDMF* algorithm and various signal degrading manipulations in order to illustrate the robustness and independence of the following performed calculation steps and furthermore the applicability of such systems in real life situations. Each of the test cases has been repeated for 10 times over the whole set of input audio files for getting decently reliable results. A complete overview of all test cases including the produced results is given by Table 5.9 in the summary of this chapter.

Figures

In the following sections the figures that show histograms illustrate the number of tests passed on the x-axis, the relative number of instances that have reached the particular number of succeeded identification attempts are shown on the axis of ordinates. Additionally there are median and mean values available. These values commonly do not appear in histograms, but they illustrate a rough meaning for the resulting accuracies and are represented by vertical lines.

5.3.2 Test case 1 - Matching single segments to database entries (RP)

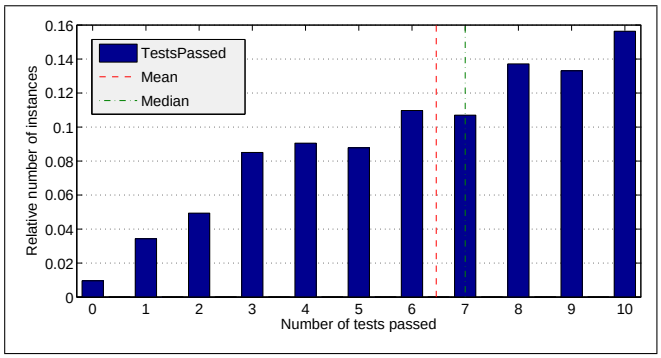

Figure 5.2: *Histogram test case 1 - Tests passed*

As mentioned before, the extraction process of Rhythm Patterns extracts feature values for several segments with specified size. The feature values are representatives for modulation frequencies at a number of specific critical bands. (compare [Lid06, page 31 ff.]) In order to find out, which window length is needed for a reliable identification and therewith acceptable precision, the first case tests single segments of the fixed size of 2^{18} samples (which corresponds to 5.94 seconds) against the description of the whole files in the database. This is done because we want to find out how stable an identification can be achieved despite a quite small analysis window and depending on the location of the start position. For each music file random sections are analysed. This step is repeated 9 times, varying the regarded position to simulate real world situations where a fingerprint has to be extracted

and calculated reliably anywhere in a file or audio stream. Figure 5.2 shows the relative amounts of tests passed in a histogram. Speaking in terms of likelihoods, this implies that the probability that an identification is done accurately is stated by a percentage of about 64.55%[6], which in fact is the mean value of tests passed in relation to the overall amount of tests and is called the precision (referred to as P_p). The first row of Table 5.3 indicates the precision values for this test case with a window of the size of one segment.

The Rhythm Patterns for the references in the database have been extracted out of whole files. The number of considered segments depends on the specific length of the file but the algorithm tries to divide the tracks into as many 2^{18}-samples-parts as possible. These values are summarized by calculating the median and treating it as description for the whole file.

5.3.3 Test case 2-3 - Increasing the number of considered segments (RP)

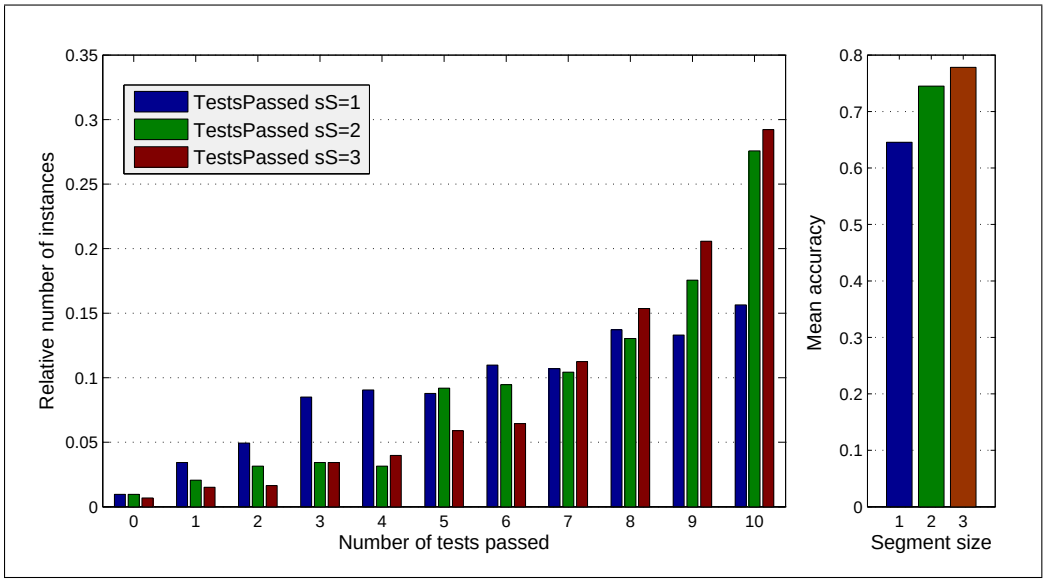

Figure 5.3: *Histogram test case 2 - Tests passed - variable segment size*

[6]Due to random selection and averaging this result may vary when reproducing the scenario, but it is considered as an admissible number for identification quality

Keeping an eye on the extraction process for a Rhythm Pattern it seems to be obvious that an increase of the partial audio signal query given as input should consequently result in less distance between the query and the reference data in the database. Taking multiple segments and applying the median should increase the similarity between the observed and the pre-recorded signal.

Furthermore, distance values of pairs which do not represent equal contents increase, which diminishes the precision and leads to worse relevance of result during the identification procedure. Multiple segments are extracted and stored in a list with the goal to apply summarization techniques on them. In this test the feature extraction of the single segments are transformed into a single Rhythm Pattern representation by applying the median function.

During the experiment characteristic values have arisen and are listed in Table 5.3 as well as they are illustrated in Figure 5.3. From left to right, the relative numbers of instances per passed test run are visible. Each colored bar represents a specific segment size (1-3). The right sub-figure shows the mean values which constitute the accuracy for each scenario.

Segments	T_t	T_p	T_f	P_p	$P_\pm$
1 ($\times 2^{18}$ samples)	7290	4706	2584	64.55%	-
2 ($\times 2^{18}$ samples)	7290	5431	1859	74.50%	+15.41%
3 ($\times 2^{18}$ samples)	7290	5673	1617	77.82%	+4.46%

Table 5.3: *Test case 2-3 - Results of variable segment size* [7]

From these results the statement can be made that more input data actually induces a more precise recognition of audio data using *Rhythm Patterns*. Increasing the segment size by a factor of 2 generates an improvement of precision by 15.41%. The utilization of a segment size of 3 times 2^{18} samples again results in better precision. Compared with the single segment results it is stated by a relative increase of 20.55% in precision, using 2 segments as the reference, the gain of precision is given by 4.46%. Of course, the best results can be achieved when matching as much reference audio as possible against the database entries. A comparison of totally equal contents of

[7] Identifiers: $T\ldots$ Tests: $_t$ total; $_p$ passed; $_f$ failed; $|\,P\ldots$ Precision: $_p$ passed; $_\pm$ relative increase

a whole piece of music leads to minimum distances towards zero. Non-zero values just occur due to rounding inaccuracies, limitations of the used data-types and partially due to different offsets in time. The resulting accuracy is probably inferior on larger databases, but nevertheless achieves almost 78% on the test database using 3 segments (i.e. approximately 18 seconds) as input for retrieval. This characteristic value will be used in the following test cases as a reference for precision.

5.3.4 Test case 4-7 - Variable segments size and whole files (FDMF)

Since there is the goal to compare RP and FDMF to find out which fulfils our needs best to enable a reliable identification system, the same tests that have been done for Rhythm Patterns are applied to the Find Duplicate Music Files-algorithm now.

The FDMF algorithm has been designed to fit the requirements of a system which only determines whether files are equal concerning the contents or not. And that is what it does according to the results of the elaborated tests. There is no information and therefore no reason for drawing conclusions about any similarity between query and reference. Test case 7 performs analysis for whole tracks which leads to a mean accuracy of $\approx 100\%$. FDMF takes all bytes into account and processes them exactly in one-second steps. It just drops some of them at the end of an input file which represents less than 1 second. The comparison of the observed audio data is still done to the pre-extracted FDMF-signatures which originate from whole tracks.

Segments	T_t	T_p	P_p	$P_{\pm}$
1 ($\times 2^{18}$ samples)	7290	11	0.15%	-
2 ($\times 2^{18}$ samples)	7290	16	0.22%	+45.45%
3 ($\times 2^{18}$ samples)	7290	23	0.32%	+43.75%
all (full track)	7290	7289	99.99%	-

Table 5.4: *Test case 4-6 - Results of variable segment size*

Processing audio data that is utilized as query of only 1, 2 or 3 segments (each of about 6 secs. of length) results in minimal accuracies of less than 1% (i.e. test cases 4, 5 & 6, cp. Table 5.4). The precision only increases if descriptions of the observed signals are made for the same length as it has been done when pre-recording and

filling the database. A proper identification testing segments against whole tracks as references can not be achieved using the FDMF as is. There are more intelligent solutions such as extracting single segments and storing them in a list like manner to the database maintaining the information about the time occurrence of the regarded part. These extracts can then be compared incrementally to both find the position in time which can be used for comparison purposes and in addition to that the corresponding piece of music [HKO01]. Such tests have not been performed in this thesis, but can be comprehended in [HKO02].

Figure 5.12 makes clear that the threshold for a precision value that would be acceptable lies between 90% and 100% of amount of query signal data in relation to the overall length. Comparing this to common identification systems that process only some seconds (cp. Chapter 3.2.3) lets claim that FDMF cannot be used in present form for our purposes.

For this reason further test series which should be performed for both RP and FDMF have been omitted for FDMF which can be noticed by the missing results in Table 5.9. In general, signal modifications, noise addition and others produce worse results than the original, just cropped query signal. It is the same here and therefore there is no need to list the results which lie below 1% of precision.

The following chapters thus deal with the RP approach only and investigate the reliability and robustness in identification processes.

5.3.5 Test case 8 - Majority Voting (RP)

Another way of evaluating the results of the whole extraction process alternative to combining or summarizing the single segment results by median is the majority vote technique. In this case, 3 segments are analysed independently and are not summarized by sum, median or mean values. For each individual segment the identification is done and the resulting distances are saved individually. The track which gets most best-matches wins and is regarded as *the* identified audio file.

Using this technique, the accuracy is narrowed down to 72.76%. This happens because 3 segments seem to be not enough to get a clear decision or eliminate outliers.

If the results of individual parts do not obtain an absolute majority, there is a fallback available that enables the usage of the calculated median values beforehand to encourage a clear decision. In 2044 out of 7290 or 28.04% of the cases, the 3 segments deliver 3 different identification results and the fallback has to be applied. More segments may induce higher precision, but there is still the requirement that as few input data as possible should be needed for an identification process. Better performance can be achieved by using more single analysis steps and / or adding another criteria (e.g. summarizing, reduction, combination with other features, etc.) for decision when getting too many different results.

5.3.6 Test case 9 - One bit median quantization (RP)

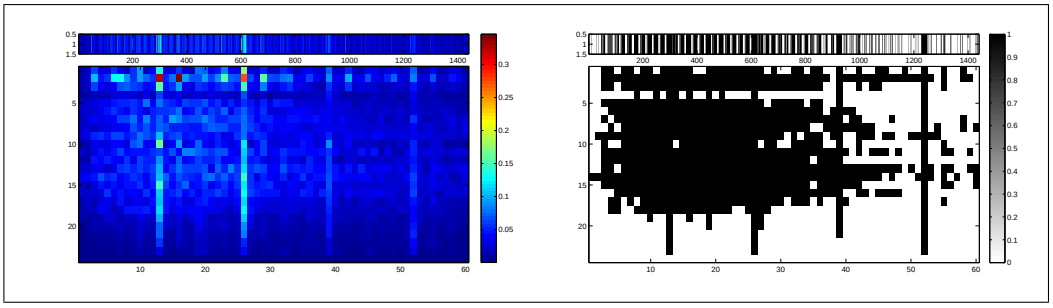

Figure 5.4: *Conversion from decimals to bits*

As mentioned in the theoretical part of this thesis (Chapter 3, it is a common technique to build bit representations out of decimals to get a fingerprint. This can be simply done by several methods. In this case, I have chosen a one bit median threshold quantization, which is characterized in equation 5.2.

Quantization for each instance

One possibility for using this approach is to quantize every query after extracting the RP as normal. The result of this procedure is a bit sequence containing the same number of bits as the corresponding vector contains double values. The database is also quantized with the same procedure. Any kind of distance measurement or

comparison may be easily performed by hamming distances or - in other words - bit errors. Equation 5.2 provides a formal definition of the quantization.

$$\forall(x_i \in X)\ldots b[i] = \begin{cases} 1 & \text{if } x_i > median(X) \\ 0 & \text{else} \end{cases} \tag{5.2}$$

Figure 5.5 shows the results of test case 9 - the histogram of the number of file instances per number of passed tests, 10 test runs as well as median and mean value. The abscissa shows the amount of tests passed, the ordinate indicates the corresponding relative count.

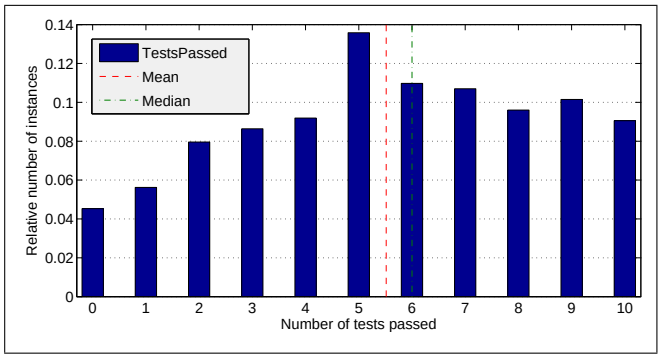

Figure 5.5: *Histogram test case 9 - Tests passed - One bit quantization*

Quantization with median RP

Another version of information reduction has been done by calculating the median values for each feature attribute over the whole test set. The result is one median feature vector which is used for the quantization step mentioned above. The main disadvantage here is, that the median value of a single median value computed from all attributes of a RP may not be representative enough as to be used as a reference. The only information which is extractable then is the bit-error distance to the median of all files, not a specific summarized distance between all features. Because of the fact that a test set should not be too homogeneous, this kind of bit sequence

comparison, which is just a possible representation for discrimination, is not suitable as the resulting values prove.

Due to such severe information reduction, a precision of just 55.16% is achievable when using the instance-based bit quantization, applying the median-RP-quantization, this is narrowed down to 51.44%. The first row of Table 5.5 provides details.

5.3.7 Test case 10-11 - Increasing median quantized segments (RP)

In the next step, the same tests which have been performed for the decimal values in test case 2 are applied to the bit representations of the Rhythm Patterns. Although this severe reduction of the available information by the bit quantization, the identification accuracies have only slightly declined. Doubling the query signal length results in a precision of 65.65%, considering 3 segments of audio, this can be extended again and amounts 69.76%. (compare Table 5.5)

Segments	T_t	T_p	T_f	P_p	$P_\pm$
1 ($\times 2^{18}$ samples)	7290	4021	3269	55.16%	-
2 ($\times 2^{18}$ samples)	7290	4786	2504	65.65%	+19.03%
3 ($\times 2^{18}$ samples)	7290	5086	2204	69.76%	+6.27%

Table 5.5: *Test case 9-11 - Varying the segment size considering median quantized vectors*

Having a look at Figure 5.6 it becomes clear that the reliability of the identification process is growing with the number of segments and both the median and mean values are increasing. Remember that all segments are extracted at random positions.

5.3.8 Test case 12-14 - Retrieving ranked results (RP)

Until now the tests have only been defined as *passed* when the algorithm has been delivering the right tracks with minimum distance (i.e. the top position in terms of similarity). In real world scenarios this way of result delivery could be varied. Users may want to retrieve a result set with a possible list of potential candidates that could match the query even if the distance is not minimal. Therefore test cases

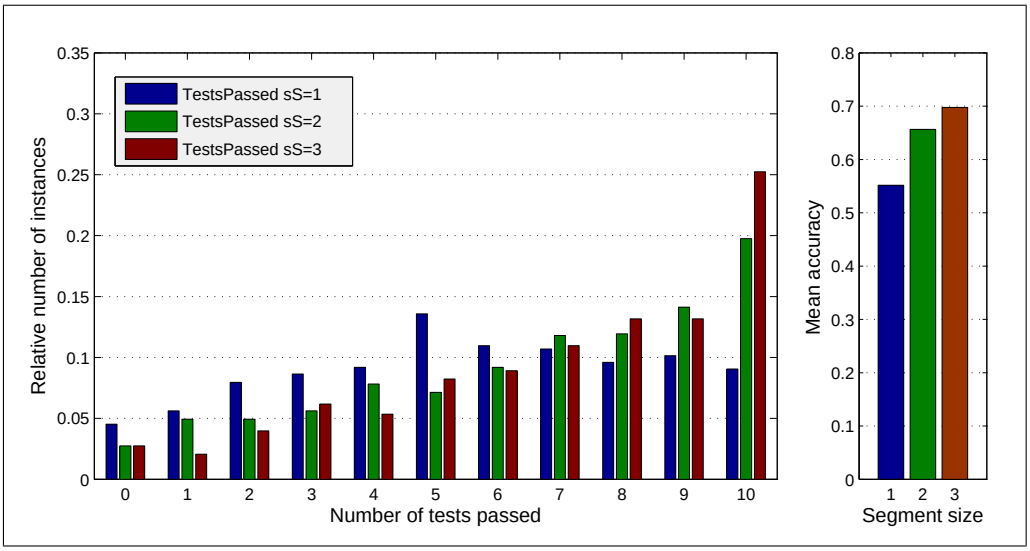

Figure 5.6: *Test case 10-11 - Median quantized values*

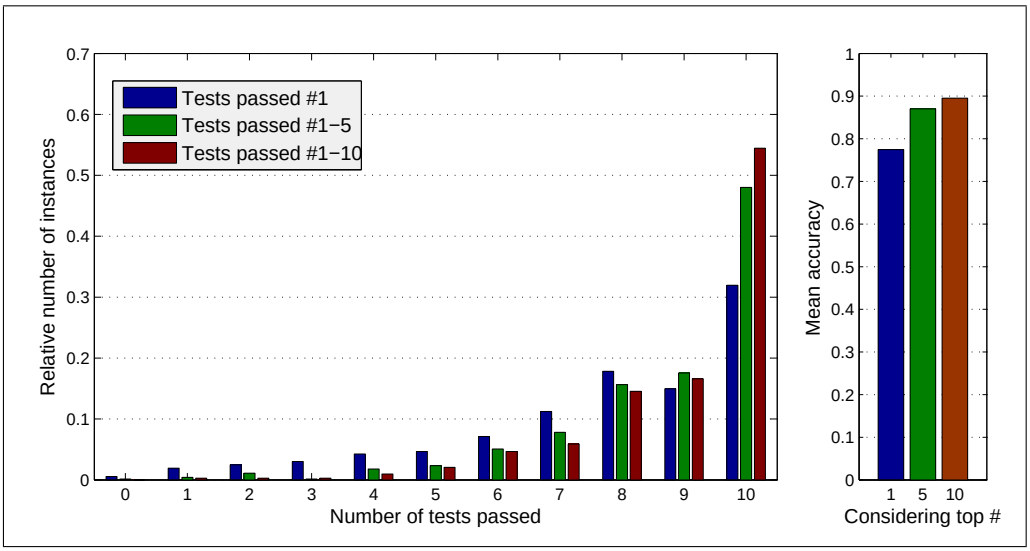

Figure 5.7: *Test case 12-14 - Retrieving ranked results*

12-14 returns all database entries with specified similarity to the reference audio. Weakening the requirements in a way that the first 5 or 10 results are considered to be *recognized* as well results in better identification accuracies. In so far this is an acceptable solution because users might find the right answer to their query in the top 5 or 10 results as well. This seems to be useful because the system may return wrong results and then the second or third match (etc.) could be the right result. Additionally providing more than one candidate includes helpful information for users which are interested in related or similar tracks. Even though a statement about similarity is not the aim of an identification system, this is an appropriable side effect.

Rank interval	T_p	P_p	$P_{\pm}$
1	5673	77.82%	-
1 - 5	6344	87.02%	+11.82%
1 - 10	6527	89.53%	+2.88%

Table 5.6: *Test case 12 - Ranked results in numbers*

As evident from Table 5.6 the recognition rate can be increased up to $\approx 90\%$. All $\pm$-percentages are given relatively to the 3-segment precision of test case 2: 77.82%. Figure 5.7 illustrates the gain of reliability when increasing the range of the interval in which an identification is considered to be successful. The x-axis indicates the number of tests passed, where the y-axis shows the relative number of songs - the precision. Additionally, the mean values for 3-segment identification accuracy considering different numbers of top positions are given by the right sub-figure.

5.3.9 Test case 15-20 - Pitch cue variations (RP)

The following test cases will investigate the robustness of the *Rhythm Patterns* algorithm for several signal modifications starting with varying the playback speed.

As opposed to time stretching I want to simulate a real world scenario like the pitch adjustment control of an analog turntable which in fact is a quite common technique for seamless beat matching of 2 concurrently running tracks. Hence, a robust audio identification system should be able to work nearly independent of

decent pitch variations. [ECM08] Rhythm Patterns do completely the contrary, but for small variations, the recognition numbers are satisfying. Variations of the original signal by the factor ± 0.01 (corresponds to 1%) result in a decreased detection rates by -1.36% / -7.65%. Table 5.7 gives an overview. The relative percentages ($P_{\pm}$) again have been compared to the mean identification precision based on the consideration of 3 segments as described in test case 2 (77.82%).

Pitch cue	T_t	T_p	P_p	$P_{\pm}$
-8%	7290	1374	18.85%	-75.78%
-3%	7290	3042	41.73%	-46.38%
-1%	7290	5239	71.87%	-7.65%
$+1\%$	7290	5596	76.76%	-1.36%
$+3\%$	7290	4000	54.87%	-29.49%
$+8\%$	7290	1957	28.85%	-65.50%

Table 5.7: *Test case 15-20 - Pitch cue variation results*

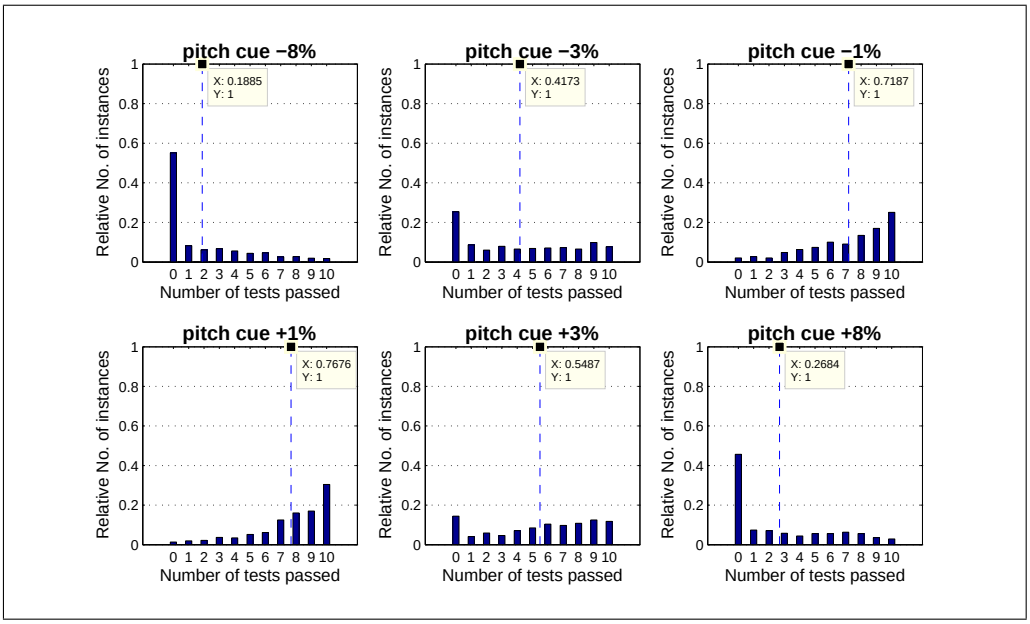

Figure 5.8: *Test case 15-20 - Pitch cue variation*

In Figure 5.8 there is a more explicit presentation of the results. The horizontal

axis again indicates the number of passed tests, the ordinate shows the precision for given tests and pitch cues. Ascending envelopes are provided by ±1%, all others have a relatively high number of unrecognized music tracks (large bars at tests passed 0%). Generally, there seems to be better recognition when increasing the speed rather than decreasing. The resampling step has been done by linear interpolation algorithm which is quite satisfying in cases of small pitch variations.

5.3.10 Test case 21 - Frequency filter (RP)

A common use case for music identification which has been implemented and has already been used by customers of mobile telephone networks is a transmission of the audio material through the mobile phone system *Global System for Mobile Communications* - GSM. MP3-encoded files are able to contain frequencies up to $22050Hz$. Using GSM technology, frequencies just less than $4000Hz$ are reproducible, but in fact this is narrowed again by interferences and low quality audio devices both on the sending and receiving side. For simulation a band pass filter of type butter-worth with order 10 has been used for limiting the signals. All frequency lots above $3000Hz$ and below $300Hz$ have been eliminated by filtering the query signal. The reference instances stay unfiltered.

Tempo or rhythm information as drums, bass lines or beat sounds are located in very low frequency bins in many cases. So it's not astonishing that Rhythm Patterns will suffer from this filtering experiment, which is expressed by the decreased accuracy of 39.84% which is quite a huge loss of reliability, compared to the 77.82% of test case 2 where no filters have been applied and the full frequency range has been maintained. Keeping an eye on the filtered Rhythm Patterns makes clear that the bark scales below $z = 3$ $(20Hz - 300Hz)$ and above $z = 17$ $(3150Hz - 15500Hz)$ have been eliminated. So a frequency range reduction like this is leads to a loss of the influence of these bands. A representation illustrating the complete bark scale is given in Chapter 3, Figure 3.1.

5.3.11 Test case 22 - Dynamic range compression (RP)

In many cases broad casted material is being modified by a dynamic range compression when finalizing the stream to offer it to others. A compression in terms of the dynamic range means that the amplitude of an output signal is compressed when the input signal exceeds a fixed threshold value (e.g. dB-value). The effective procedure is defined by the parameters as follows:

- Compression ratio: $4:1$
  the compression ratio which is applied to signal elements that exceed the threshold;
- Threshold: $-12dB$
  threshold for compression criteria;
- Attack time: $300ms$
  the time span in milliseconds which is needed for the system to react on the compression criteria;
- Release time: $2000ms$
  time span in milliseconds where the compression effect is dying away. After the release time the signal is equal to the original if no new threshold exceeding occurs;
- Knee: hard knee
  hard knee means that there is a sharp angle in the bend of the response curve instead of a rounded edge (soft knee);

See Figure 5.9 for an illustration of the compressor that has been used.

The results of the analysis of the compressed files does not worsen the resulting numbers aggravating. By contrast with the original tracks there is a precision number of 72.57% at 3 segments. Remember that the original, uncompressed files have been identified in 77.82% of the cases - so the compression step caused a reduction of precision of 6.75%.

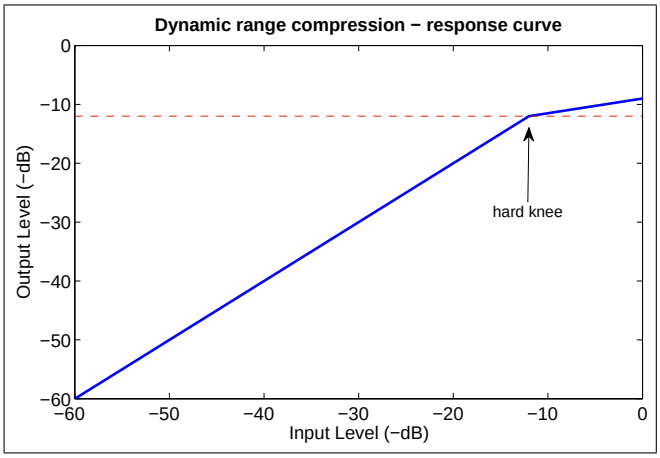

Figure 5.9: *Dynamic range compression - Response curve*

Noise	T_t	T_p	P_p	$P_{\pm}$
White noise	7290	5661	77.65%	−0.21%
Pink noise	7290	5665	77.71%	−0.14%

Table 5.8: *Test case 23-24 - Addition of white / pink noise*

5.3.12 Test case 23-24 - Addition of white and pink noise (RP)

Processing signals which are recorded or created by analog devices or media usually are distorted by some kind of noise. Even if it is not perturbing or perceivable for human beings as a fence for identifying music, automatic solutions can be dazzled by too much addition of noise signals. The definition of *noise* can be subdivided into many colors (black, blue, brown, gray, green, orange, pink, purple, red and white) and refers to a bias which tends to a specific range of frequencies. The most popular noise signal is white noise, which has a regular deviation of frequency lots (f-noise). Pink noise ($1/f$-noise) has narrowed energy for increasing frequencies in the spectral representation. The deviation of brown or $1/f^2$-noise is in squared inverse proportion to the frequency bins. Figure 5.10 depicts the spectrogram for both white noise (left) and pink noise (right). The color map along the y axis shows the narrowing energy for growing frequency bins.

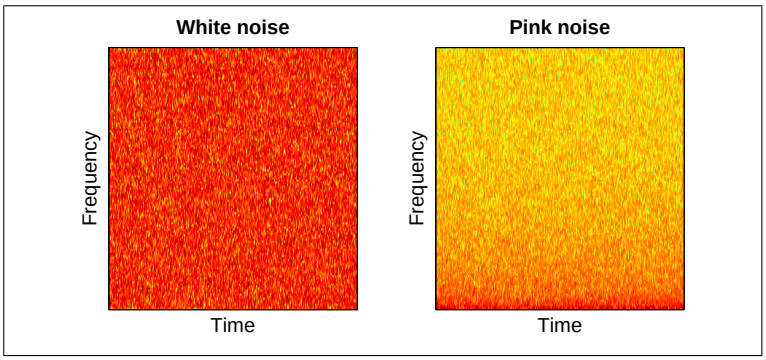

Figure 5.10: *Comparison of white noise and pink noise*

The tests show that the addition of white noise with the intensity of 0.1 decline the precision results to 77.65% in comparison to the analysis of 3 segments of the original signal. All tests using pink noise as disturbing signal result in a mean accuracy of 77.71% (compare Table 5.8. In terms of an audio signal this intensity is a value compared to a normalized maximum amplitude of 1.0. This corresponds to a SNR (Signal-to-noise ratio) of $20dB$ which is clearly remarkable for humans. Equation 5.3 illustrates the formal definition for signal-to-noise ratio (SNR) computation.

$$SNR = \left(\frac{Amp_{signal}}{Amp_{noise}}\right)^2$$
$$SNR_{dB} = 10 * \log_{10}\left(\left(\frac{Amp_{signal}}{Amp_{noise}}\right)^2\right) \tag{5.3}$$

5.4 Summary

Table 5.9 provides an overview of the performed tests and results. The third column indicates the best result of the test case or of the series which is covered by the test case number.

What I actually wanted to figure out by this practical part of my thesis is the difference between feature-based similarity measurement identification / retrieval techniques and audio fingerprinting techniques using Rhythm Patterns and FDMF as exemplary algorithms.

Rhythm Patterns are very useful when having small chunks of audio data and comparing them to the extracted features of complete files. Much better reliability would be achieved when comparing to segments of the same size as it has been tested by full music tracks and reduced database sets, but this would consequently lead to much higher segment search and alignment times. I have shown that there must be and there actually is a higher identification precision when increasing the considered query audio samples as the objective similarity gains.

By contrast fingerprint algorithms generally map audio data onto binary string representations. This means that a very small change of the signal can result in totally different bit strings. FDMF is quite intelligent because of the fact that it processes frequency bands and comparing them to each other, but it is just acceptable when having large audio signal parts at least of 90% of the original length.

Figures 5.11 and 5.12 illustrate very straight how the precision values vary in the different procedures. RP has its biggest slope at less than $(0.1 * meanFileSize)$. For the rest, the more data is available the more likely is a identification in linear manner. The mean detection of FDMF has little accuracy until almost $(0.9 * meanFileSize)$. The fact that the curve rises at 50% just accrues because of the small size of the test

Test case #	Description	add.	$P_p(RP)$	$P_p(FDMF)$
1 | 4	Matching single segments		64.55%	0.15%
2 | 5	Increasing the segment size	$sS = 2$	74.50%	0.22%
3 | 6	Increasing the segment size	$sS = 3$	77.82%	0.32%
7	Analysing full tracks		100.00%	99.99%[a]
8	Majority vote		72.76%	–[b]
9a	One bit median quantization	per RP	55.16%	–
9b	One bit median quantization	per attribute	51.44%	–
10	Increasing median quantized segments	$sS = 2$	65.65%	–
11	Increasing median quantized segments	$sS = 3$	69.76%	–
12	Retrieving ranked results	[1]	77.82%	–
13	Retrieving ranked results	[1..5]	87.02%	–
14	Retrieving ranked results	[1..10]	89.53%	–
15	Pitch cue variation	-8%	18.85%	–
16	Pitch cue variation	-3%	41.73%	–
17	Pitch cue variation	-1%	71.87%	–
18	Pitch cue variation	$+1\%$	76.76%	–
19	Pitch cue variation	$+3\%$	54.87%	–
20	Pitch cue variation	$+8\%$	28.85%	–
21	Applying frequency filters		39.84%	–
22	Dynamic range compression		72.57%	–
23	Addition of white noise		77.65%	–
24	Addition of pink noise		77.71%	–

Table 5.9: *Test cases 4-6 (FDMF), test cases 1-3, 8-24 (RP) - Overview*

[a] These values may differ from 100 percent due to rounding inaccuracies and limitations of the underlying data-types.

[b] Further tests on FDMF were omitted due to the low performance of test cases 4 to 6 being < 1 %

set which has been used for the experiments. Performing the tests for large sets of many thousands would result in a similar small slope for x-values 0.4 to 0.9 like it occurs in range 0.1 to 0.4.

Rhythm Patterns are definitely the better solution for the requirements I am concentrating on. The real-time analysis of streamed data is always processed for audio snippets, not for whole files. FDMF as is will stay a duplicate file finder if it's not altered in some way that concentrates on music information not on file content information.

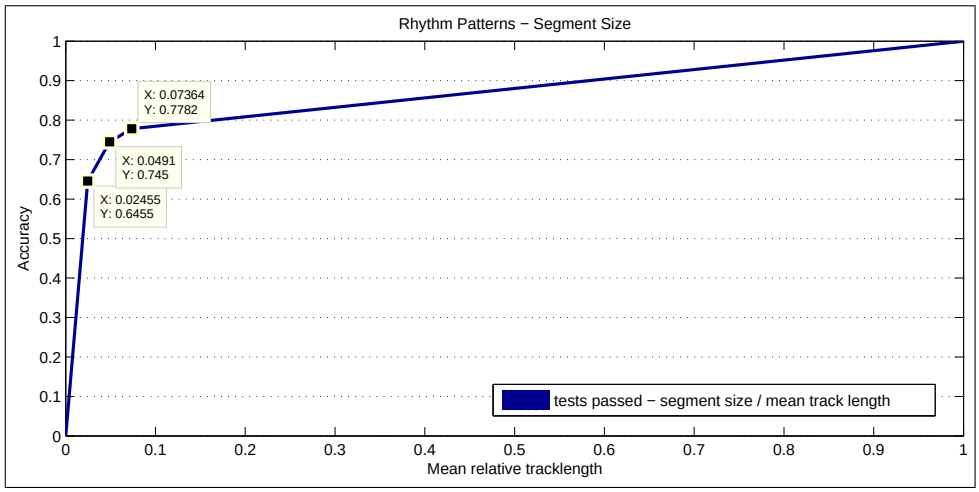

Figure 5.11: *Various segment size - Rhythm Patterns*

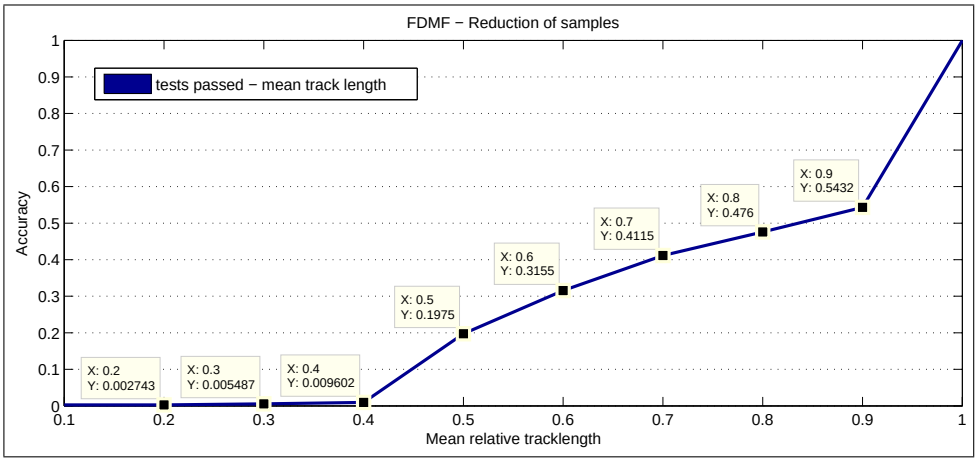

Figure 5.12: *Various segment size - FDMF*

Chapter 6

Implementation Details

This chapter is about the implementation part of my thesis. Several components concerning the extraction of the fingerprints or similarity features as well as capture classes and a graphical user interface (GUI) have been developed and are described here. I will give a brief overview about the functionality and the used architecture. A detailed documentation can be found online on the complementary web site to this thesis.

6.1 Functionality

The implementation covers the following topics:

- Audio capturing both via line input / microphone and streamed data over networks
- Extraction and computation of the FDMF-fingerprint and Rhythm Patterns [1]
- Comparison of the extracted values to the database values
- Retrieval of results, i.e. identified title(s)

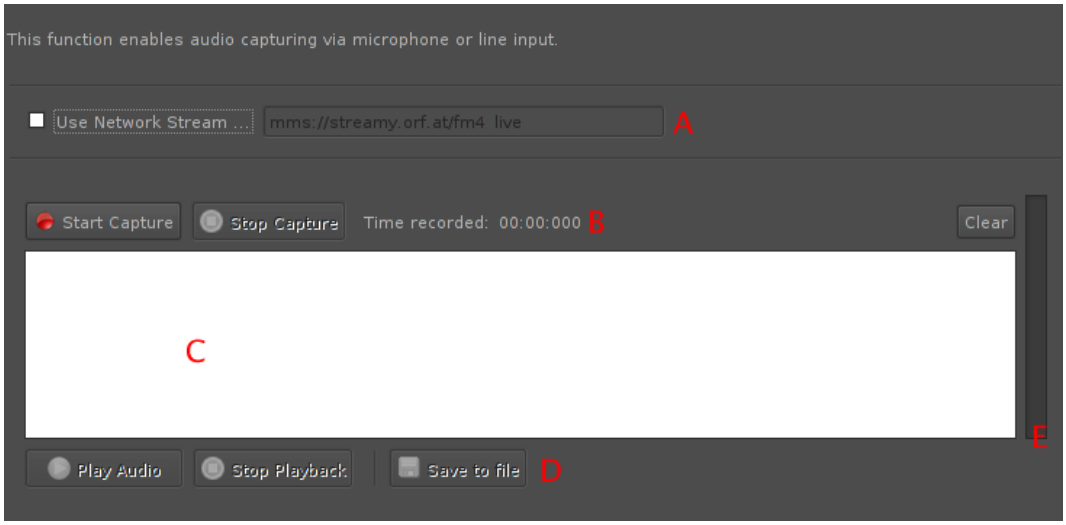

Figure 6.1: *Graphical user interface - Capture panel*

6.1.1 Capturing Audio

There are two main approaches and therewith two different views how to capture audio data both over network based streams and local audio hardware. Figure 6.1 shows a screen-shot of the panel with GTK (Gimp Tool Kit[2]) Look and Feel. The GUI provides a simple check box to determine whether to use network or local audio hardware (mark **A**). Owing to the implementation of a Java interface to MPlayer, it is possible to insert any URL here which can be processed by MPlayer. In addition to that the user interface provides audio controls to start and stop the capture (mark **B**) as well as a timer that indicates the amount of audio data captured in terms of time. Furthermore there are a list which shows the already captured audio snippets (mark **C**), the controls for starting and stopping the playback (mark **D**) and a sound level meter (mark **E**).

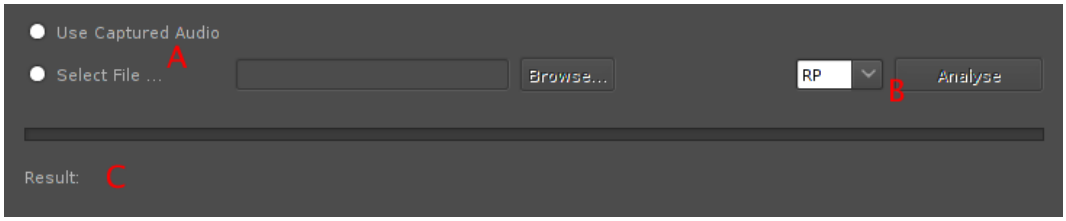

Figure 6.2: *Graphical user interface - Analysis panel*

6.1.2 Computing Fingerprints and Features

Figure 6.2 indicates the possibilities for further processing. Having the audio data recorded, the capture class saves the recorded items to a list which is displayed and can be selected to play it back, save it or analyse it (mark **A**) using FDMF or Rhythm Patterns (mark **B**). Additionally, a local file can be selected as well to be processed via the analysis algorithms. This feature can be used to perform fundamental tests on audio files with known contents or on externally recorded signals.

6.1.3 Retrieving Results

The procession steps of extraction, computation of features and comparison to entries in the database are totally transparent and their current progress can just be monitored through a single progress bar which shows the overall course. The result of these processes is simply given by one database key which fits best to the given query (mark **C**) according to the distance metrics which have been implemented (Euclidean for Rhythm Patterns, bit error count for FDMF).

6.2 Architecture

In this section I will not describe the whole architecture in detail including classes, libraries, methods (...) but I will give a brief illustration of the core elements, how

[1] Partially integration of existing libraries
[2] URL: http://www.gtk.org/

they work together and in what sense they are extensible and exchangeable by new or different realizations.

6.2.1 Architectural Overview

The following class diagrams (Figure 6.3 and Figure 6.4) are simplified for a more schematic representation and do not cover all elements according to the full-fledged entity set of the Unified Modeling Language (UML).

Extraction and Comparison

Basically, there is one main JAVA interface which predetermines the structure of the implementation classes - *ICompare*. Classes that implement this interface have to provide at least the methods which are necessary for a proper return of the found database keys when passing either an audio byte array or nothing, when the data has already been set by another way. To ensure that a comparison can be accessed only once in a moment, singleton constructors are used. Additionally, there are two classes for feature extraction or fingerprint computation which are invoked by the *RPCompare*- and *FDMFCompare*-classes - the extractors. For extension, it is just essential to implement *ICompare* as interface and to write the relevant code which extracts and compares the given prototype data to the database. Finally there must be a return of the adequate song title.

Audio

The second functional part which I will describe concerns the capture, playback, storage and conversion of audio data (compare Figure 6.4). As already mentioned, there are two possible ways how to record audio signals and thus they are covered by an interface - *ICapture*. The first one uses the local audio hardware to capture either via a connected microphone or the line in jack. This is done using the Java Sound API, respectively using the Java Sound Resources [3], a Java Sound Implementation.

[3] http://www.jsresources.org/

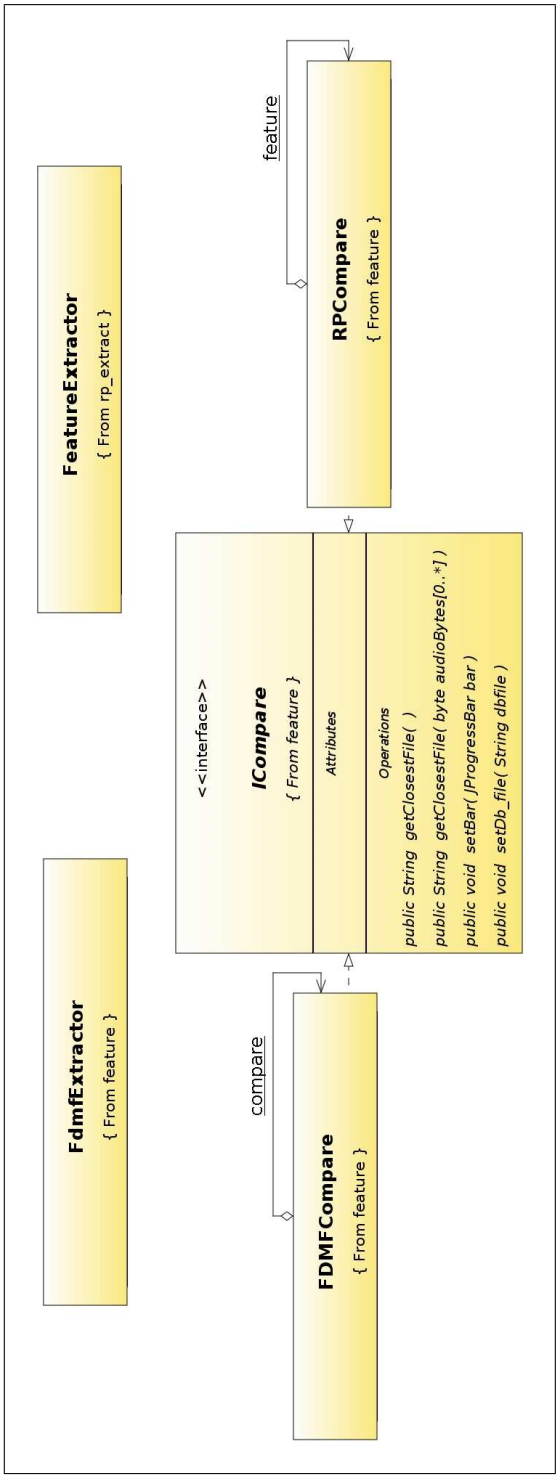

Figure 6.3: *Class diagram for feature extraction and comparison*

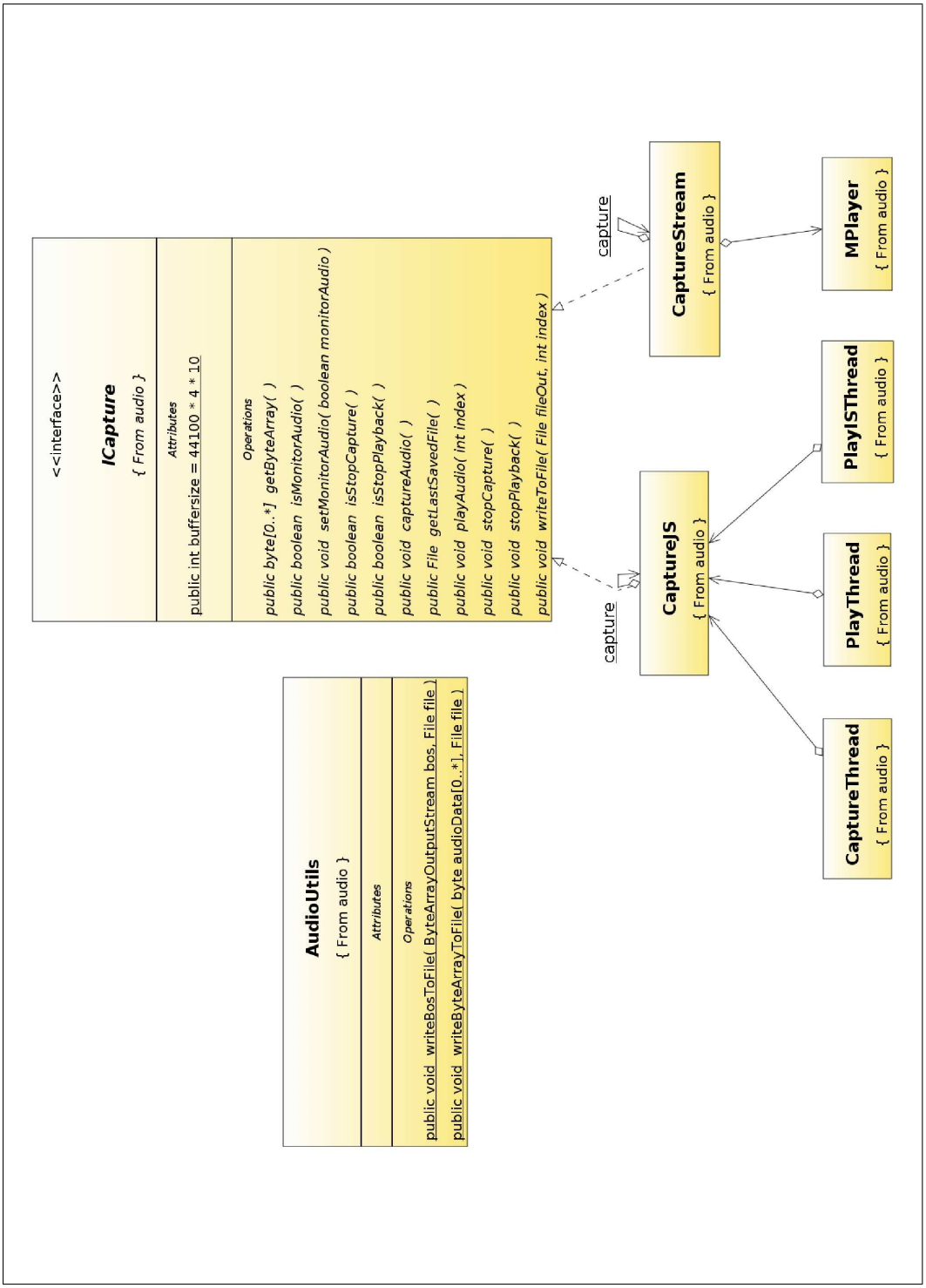

Figure 6.4: *Class Diagram for audio capture, playback and storage*

The class *CaptureJS* provides the full functionality for both record and playback. There are two Threads (*CaptureThread* and *PlayThread*) which work independently. These Runnables are started and stopped via simple Boolean flags by the *CaptureJS* class which thus serves as a controller. Using this structure enables the possibility to capture, store and play the incoming signal concurrently. For seamless access and avoidance of data loss, InputStreams [4] are utilized. The third Thread (compare Figure 6.4) is used for offering several methods to external components, which just want to use the functionality of *CaptureJS*, but provide an own InputStream which should be accessed.

On the other side there is the requirement which has led to the possibility to work with network based streams as easy as using local audio. Thus the interface provides the same methods and fields to the *CaptureStream*-class. By contrast to *CatureJS*, no Threads are used, but an interface class for MPlayer, an audio player for Linux, has been developed. In this case, real processes are started which perform all necessary actions on the audio source URL. The big advantage of using MPlayer for capturing streamed data is that there is a quite large amount of features available which are already implemented and can just be used as they are. To preserve the consistency to the JavaSound approach, there are local stream objects to guarantee access to the data. Here, the output of MPlayer is redirected to the standard output stream (STDOUT), which can be read and processed by other operations system processes. Using Java, the STDOUT can be accessed via InputStream / OutputStream objects as desired.

The class *AudioUtils* just provides static functions which can be used by all other components. These methods cover conversion steps of audio from Java object representations like byte arrays or buffered streams to a file format that can be read by common applications.

For further information as documentation, source code, or the application itself have a look at the complementary web page.

[4] http://java.sun.com/j2se/1.5.0/docs/api/java/io/InputStream.html

Chapter 7

Conclusion and Outlook

7.1 Summary and Review

Finding recapitulating words to finalize this thesis and therewith the comparison of feature-based similarity measurement techniques and real fingerprint approaches as well as the general outline of the underlying theory and applications will be about the usage and acceptance of such systems in real world scenarios.

Identification mechanisms enable the recognition of parts of audio both local and via network-based streaming technology. The association with a specific position in a captured piece of music must be able to deal with deteriorated signals, very small excerpts of audio tracks at a randomly selected position and the complexity of the search procedure for appropriate candidates. In chapter 4.3.2 it comes out that database-querying software components have to define clear structures of the used data fields or sub-fingerprints. Decisions which provide early information about the overall plausibility are able to result in short response times and therefore usability for non-expert users. An automatic assignment of the input track to a classification scheme can also improve the acceptance of potential community members.

As proven in Chapter 5, similarity based feature sets are usable for identification purposes. The proper selection, weighting and the comparison in a way that tries to model the human understanding for musical similarity is a possible way to indisputable recognize music or audio signals in general. Conceiving the entirety of audio tracks, respectively the numerical vector representations as a huge multi-dimensional

vector space where similarity or identity can be expressed by distance measures and metrics, a powerful approach is given to ensure that identification is a task which can be coped with.

It has come out that the FDMF-algorithm is definitely not usable for these kinds of tasks. The detailed evaluation has been omitted due to the low identification precisions of the basic test scenarios. For this reason, the results of the test cases which have been performed are given for the similarity-based approach only.

Considering small excerpts of music tracks, the Rhythm Patterns experiments have shown that identification accuracies of 64.55% for ≈ 6 seconds of audio, 74.50% for twice as much, and 77.82% for 3 segments of the same size can be achieved without any alteration of the algorithm, just by selecting the segments at random. Varying the evaluation strategy improves the precision up to 89.53% (i.e. accepting ranked results from rank #1-#10). Signal modifications have deep impact on the extraction and therewith the resulting numbers of identification quality. Pitch cue variations (i.e. tempo increase / decrease) of $\pm 1\%$ can be neglected, applying higher alterations leads to unacceptable accuracies of less that 50%. Limiting relevant frequency ranges by applying frequency filters constitute a real problem for the feature extraction, whereas the addition of noise or dynamic range compression do not influence the whole procedure severely.

7.2 Future Work

Applications in future will have to combine several techniques. These can be similarity based approaches and exact fingerprints. Another improvement could be achieved when adding intelligent classifications or clustering on the database side of the identification system. Applications will have to manage more and more audio titles as the overall availability of digital music material as well as the number of central or global administration systems gain.

Distributed systems would be a proper solution for managing the huge load of data. Externalizing the signature extraction to client computers and just realizing

the comparison and search on the server side would lead to a distributed mighty system that allows to exchange information between the user community and integrate a monitoring component. Using such an architecture would ensure that audio identification is not limited to personal computer clients but enables the use of handheld, mobile phones or integrated components into common consumer electronic devices.

In general, the possibility of identifying audio in consideration of the content only would lead to major benefits and values for people interested in music.

List of Figures

List of Tables

Literature

[AHH+03] Eric Allamanche, Jürgen Herre, Oliver Hellmuth, Thorsten Kastner, and Christian Ertel. *A Multiple Feature Model for Musical Similarity Retrieval.* Proceedings of International Conference for Music Information Retrieval (ISMIR) 2003, Baltimore, Maryland (USA) (2003).

[AP02] Jean-Julien Aucouturier and François Pachet. *Music Similarity Measures: What's the use?* In Proceedings of International Conference for Music Information Retrieval (ISMIR), Paris (France), (pp. 157–163) (2002).

[BC06] Shumeet Baluja and Michele Covell. *Content Fingerprinting Using Wavelets.* In Proceedings of 3rd European Conference on Visual Media Production (CVMP), London UK, (pp. 198–207) (2006).

[BCR07] Michael Betser, Patrice Collen, and Jean-Bernard Rault. *Audio Identification using sinusoidal modeling and application to jingle detection.* In *Proceedings of International Conference for Music Information Retrieval (ISMIR)*, (pp. 139–142). Vienna (Austria) (2007).

[BK07] Klaas Bosteels and Etienne E. Kerre. *Fuzzy Audio Similarity Measures Based on Spectrum Histograms and Fluctuation Patterns.* In *International Conference on Multimedia and Ubiquitous Engineering (MUE'07)* (2007).

[BM07] C. Bellettini and G. Mazzini. *On audio recognition performance via robust hashing.* In Proceedings of International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), Xiamen (China), volume 1:pp. 20–23 (2007).

[BPJ02] Christopher J.C. Burges, John C. Platt, and Soumya Jana. *Extracting noise-robust features from audio data.* IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '02) (2002).

[CBG03] Pedro Cano, Eloi Batlle, and Emilia Gomez. *Audio fingerprinting: Concepts and applications.* In *Proceedings of 1st International Conference on Fuzzy Systems and Knowledge Discovery* (2003).

[CBKH02] Pedro Cano, Eloi Batlle, Ton Kalker, and Jaap Haitsma. *A review of algorithms for audio fingerprinting.* Workshop on Multimedia Signal Processing 2002 (2002).

[CBMN02] Pedro Cano, Eloi Batlle, Harald Mayer, and Helmut Neuschmied. *Robust sound modeling for song detection in broadcast audio.* AES 112th Int. Conv. (2002).

[Cer07] Jose Ramon Cerquides. *A real time audio fingerprinting system for advertisement tracking and reporting in FM radio.* IEEE 17th International Conference on Radioelektronika (2007).

[CS06] Michael Casey and Malcolm Slaney. *The importance of sequences in musical similarity.* IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2006).

[CS07] Michael Casey and Malcolm Slaney. *Fast recognition of remixed music audio.* IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2007).

[Cve04] Nedeljko Cvejic. *Algorithms for audio watermarking and steganography.* Ph.D. thesis, University of Oulu (2004).

[CZW08] Bingwei Chen, Jiying Zhao, and Dali Wang. *An Adaptive Watermarking Algorithm for MP3 Compressed Audio Signals.* IEEE International Instrumentation and Measurement Technology Conference (I2MTC) (2008).

[DG06] Jesse Davis and Mark Goadrich. *The Relationship Between Precision-Recall and ROC Curves.* In Proceedings of the 23rd International Conference on Machine learning, volume 148:pp. 233–240 (2006).

[DL05] P.J.O. Doets and R.L. Lagendijk. *Extracting Quality Parameters for Compressed Audio from Fingerprints.* In Proceedings of International Conference for Music Information Retrieval (ISMIR) 2005, London (UK), (pp. 498–503) (2005).

[ECM08] Daniel P. W. Ellis, Courtenay V. Cotton, and Michael I. Mandel. *Cross-correlation of beat-synchronous representations for music similarity.* IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2008).

[FGDW06] Arthur Flexer, Fabien Gouyon, Simon Dixon, and Gerhard Widmer. *Probabilistic Combination of Features for Music Classification.* Proceedings of International Conference for Music Information Retrieval (ISMIR) 2006, Victoria, BC (Canada) (2006).

[Foo99] Jonathan Foote. *An overview of audio information retrieval.* ACM Multimedia Systems, Special issue on audio and multimedia, Volume 7, National University of Singapore, Singapore (1999).

[GDPW04] Fabien Gouyon, Simon Dixon, Elias Pampalk, and Gerhard Widmer. *Evaluating rhythmic descriptors for musical genre classification.* In Audio Engineering Society (AES) 25th International Conference, London, UK, (pp. 1–9) (2004).

[HBW+08] Perfecto Herrera, Juan Bello, Gerhard Widmer, Mark Sandler, Oscar Celma, Fabia Vignoli, Elias Pampalk, Pedro Cano, Steffen Pauws, and Xavier Serra. *SIMAC: Semantic Interaction with Music Audio Contents.* In IEEE Transactions on Audio, Speech and Language Processing, volume 16:pp. 408–423 (2008).

[HK02] Jaap Haitsma and Ton Kalker. *A Highly Robust Audio Fingerprinting System.* In Proceedings of International Conference for Music Information Retrieval (ISMIR) 2002, Paris (France), (pp. 107–115) (2002).

[HKO01] Jaap Haitsma, Ton Kalker, and Job Oostveen. *Robust Audio Hashing for Content Identification.* In Proceedings of Content-Based Multimedia Indexing (CBMI), Brescia, Italy (2001).

[HKO02] Jaap Haitsma, Ton Kalker, and Job Oostveen. *An Efficient Database*

Strategy for Audio Fingerprinting. IEEE Workshop on Multimedia Signal Processing, Georgia, USA (2002).

[HLH07] Che-Jen Hsieh, Juang-Shian Li, and Cheng-Fu Hung. *A robust audio fingerprinting scheme for MP3 Copyright* (2007).

[IIS] Fraunhofer IIS. *MP3 - Mpeg Audio Layer 3.* URL `http://www.iis.fraunhofer.de/EN/bf/amm/projects/mp3/index.jsp`

[KE04] Alfons Kemper and André Eickler. *Datenbanksysteme.* Oldenbourg Wissenschaftsverlag GmbH (2004).

[Kos02] Karin Kosina. *Music Genre Recognition.* Master's thesis, FH Medientechnik und -design, Hagenberg, Austria (2002).

[KQG04] Serkan Kiranyaz, Ahmad Farooq Qureshi, and Moncef Gabbouj. *A generic audio classification and segmentation approach for multimedia indexing and retrieval.* In Proceedings of the European Workshop on the Integration of Knowledge, Semantics and Digital Media Technology (EWIMT), (pp. 55–62) (2004).

[KSWW00] Thomas Kemp, Michael Schmidt, Martin Westphal, and Alex Waibel. *Strategies for automatic segmentation of audio data.* In Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), (pp. 1423–1426) (2000).

[Lei04] Stefan Leitich. *Digital Music Libraries - Analysis and Comparison of Feature Sets for Audio Retrieval.* Master's thesis, University of Vienna, Vienna University of Technology (2004).

[Lid06] Thomas Lidy. *Evaluation of New Audio Features and their Utilization in Novel Music Retrieval Applications.* Master's thesis, Vienna University of Technology (2006).

[LR06] Thomas Lidy and Andreas Rauber. *Visually Profiling Radio Stations.* In Proceedings of International Conference for Music Information Retrieval (ISMIR) 2006, Victoria, BC (Canada) (2006).

[LS01] Beth Logan and Ariel Salomon. *A music similarity function based on*

signal analysis. IEEE International Conference on Multimedia and Expo (ICME), Tokyo, Japan (2001).

[MB03] Martin F. McKinney and Jeroen Breebaart. *Features for Audio and Music Classification.* Proceedings of International Conference for Music Information Retrieval (ISMIR) 2003, Baltimore, Maryland (USA) (2003).

[MG06] Paisarn Muneesawang and Ling Guan. *Multimedia Database Retrieval: A Human-Centered Approach (Signals and Communication Technology)*, volume 1. Springer, Berlin (2006).

[MTB+04] Jose P.G. Mahedero, Cadim Tarasov, Eloi Batlle, Enric Guaus, and Jaume Masip. *Industrial audio fingerprinting distributed system with CORBA and Web Services.* In Proceedings of International Conference for Music Information Retrieval (ISMIR) 2004, Barcelona (Spain) (2004).

[NL99] Frank Nack and Adam T. Lindsay. *Feature Article Everything You Wanted to Know About MPEG-7: part1* (1999).

[NMB01] Helmut Neuschmied, Harald Mayer, and Eloi Batlle. *Content-based identification of audio titles on the Internet.* Proceedings of the First International Conference on WEB Delivering of Music (WEDELMUSIC01) (2001).

[Pei07] Ewald Peiszer. *Segment Boundary and Structure Detection in Popular Music.* Master's thesis, Vienna University of Technology (2007).

[RPM02] Andreas Rauber, Elias Pampalk, and Dieter Merkl. *Using PsychoAcoustic Models and SelfOrganizing Maps to Create a Hierarchical Structuring of Music by Sound Similarity.* Proceedings of International Conference for Music Information Retrieval (ISMIR) 2002, Paris (France) (2002).

[SBA06] M. Sert, B. Baykal, and A.Yazici. *A Robust and Time-Efficient Fingerprinting Model for Musical Audio.* IEEE 2006 (2006).

[Sin06] Alexander Sinitsyn. *Duplicate Song Detection using Audio Fingerprinting for Consumer Electronics Devices.* IEEE Tenth International Symposium on Consumer Electronics, 2006 (ISCE '06) (2006).

[SWS07] Klaus Seyerlehner, Gerhard Widmer, and Dominik Schnitzer. *From Rhythm Patterns to Perceived Tempo.* In Proceedings of International Conference for Music Information Retrieval (ISMIR) 2007, Vienna (Austria), (pp. 519–524) (2007).

[TEC02] George Tzanetakis, Georg Essl, and Perry Cook. *Musical genre classification of audio signals.* IEEE Transactions on Speech and Audio Processing (2002).

[Vas07] Saeed V. Vaseghi. *Multimedia Signal Processing: Theory and Applications in Speech, Music and Communications*, volume 1. Wiley & Sons (2007).

[VHAP00] Juha Vesanto, Johan Huimberg, Esa Alhoniemi, and Juha Parhankangas. *SOM Toolbox for Matlab 6* (2000).