Audio Information Retrieval (AIR) Tools

George Tzanetakis [1]
Department of Computer Science [2]
Princeton University

Perry Cook [3]
Department of Computer Science and Department of Music
Princeton University

[1]gtzan@cs.princeton.edu
[2]Address: 35 Olden Street Princeton NJ 08544
[3]prc@cs.princeton.edu

Abstract

The majority of work in music information retrieval (IR) has been focused on symbolic representations of music. However, most of the digitally available music is in the form of raw audio signals. Although various attempts at monophonic and polyphonic transcription have been made, none has been successful and general enough to work with real world signals.

In this paper we describe some initial efforts at building IR tools for real world audio signals. Our approach is based on signal processing, statistical pattern recognition and visualization techniques. We try to gather as much information as possible without attempting to perform polyphonic transcription.

A frequently ignored aspect in emerging fields like music IR is the importance of the user in building a successful system. We describe some new graphical user interfaces that accommodate different modes of interaction with the user. More specifically we describe an augmented sound editor for annotating, classifying and segmenting music and we define *TimbreGrams* a new visual representation for audio files.

Keywords: user interfaces for music-IR, principal components analysis, visualization, segmentation, classification, retrieval, audio thumbnailing

1 Introduction

Most of the work in music IR and analysis has been performed using symbolic representations like MIDI. Because symbolic representations are easy to work with and require modest amounts of processing power there has been a large history of work in this area. There are many existing tools that parse and analyze these representations. However, most of the digitally available music is in the form of unstructured monolithic sound files.

Automatic transcription systems are the bridge that can connect the world of symbolic analysis and real world audio. Unfortunately despite various efforts at automatic transcription a robust system that can work with real world audio signals has not yet been developed.

Another approach that has been proposed is Structured Audio (SA). The idea is a hybrid representation combining symbolic representation, software synthesis and raw audio in a structured way [9]. Although promising this technique has only been used in a small scale. MPEG 4 [9], might change the situation but for now and for the near future most of the data will still be in raw signals. Moreover, even if SA is widely used, converters from raw audio will still be required.

In this paper the focus is on tools that work directly on real world audio data without attempting to transcribe the music. It has been argued that most listeners do not hear individual notes when they listen to music and therefore automatic systems that focus only on music theoretic aspects are missing important aspects of human perception [7]. To distinguish from symbolic-based music IR for the remainder of the paper we use the term audio IR (AIR) to refer to techniques that work directly on raw audio signals. Obviously these signals can contain music as well as other types of audio like speech and sound effects.

An overview of the currently available AIR techniques and how they are supported in our system is given. For an overview of related work refer to [2]. When designing an AIR system it is important to think about the user and the corresponding domain. For example query-by-humming works for classical music and simple folk songs but would not be effective with rap songs. Our main effort is to provide tools for working with the timbral and temporal aspects of sound rather than with music theoretic descriptions. Due to the immature state of the available techniques and to the inherent complexity of the task it is important to take advantage of the human user in the system. Two new user interfaces for AIR are proposed and described in detail. The first is an augmented sound-editor with automatic classification, segmentation and retrieval capabilities and the second is *TimbreGrams* a novel graphical representation for sound files. The previously unpublished contributions of this paper are the genre classification method, the segmentation-based retrieval and audio thumbnailing, and the definition of the *TimbreGram.*

2 Overview

In section 3, a synopsis of feature-based audio analysis is given. We describe classification, segmentation, retrieval and audio thumbnailing which are important elements of any AIR system. Specific examples of these processes from MARSYAS, our framework for audio analysis are given. In section 4, we describe the augmented sound editor under which all these techniques are integrated. *TimbreGrams* a novel graphical representation for sound files based on visualization techniques is described in section 5. The paper ends with some notes on the implementation and future work.

3 Feature-based audio analysis

The basis of audio analysis algorithms is the calculation of short-time feature vectors. The signal is processed in small chunks so that the signal characteristics are relatively stable. For each chunk some form of spectral analysis is performed and based on that analysis a vector of feature values is calculated (see Figure 1). These feature values are a summary description of the corresponding spectral content of the chunk.

Many different features have been proposed in the literature. Our system supports features based on :

- FFT (Fast Fourier Transform) analysis
- MPEG filterbank analysis [15, 8]
- LPC (Linear Predictive Coding) [6]
- MFCC (Mel-Frequency cepstrum coefficients) [3]

In the majority of the literature in audio analysis a combination of these types of features is used. Another frequently used technique is to compute the means and variances of these features over a larger time window to obtain more smoothly changing features. In addition derivatives of the features (or delta-features) are frequently computed to express temporal changes. Sometimes log-transformations of the features are used to reduce the dynamic range. All these techniques are supported in MARSYAS and new features can be added to the system with minimal effort.

3.1 Classification

Statistical pattern recognition refers to a series of techniques where the extracted feature vectors are assigned to one of **c** classes. Classification algorithms are divided into unsupervised and supervised. In the supervised case a labeled set of training samples is provided that is used to "train" the classification algorithm. Unsupervised classification or clustering tries to group the data into meaningful clusters without the use of a labeled training set. Another way of dividing classification algorithms is parametric vs non-parametric algorithms. In parametric approaches the functional form of the underlying probability destribution of the feature vectors for each class is known. In non-parametric approaches no assumption about the functional form is made and the underlying probability distribution is approximated locally based on the training set. For more details see [1].

In MARSYAS, the Gaussian (MAP) and Gaussian Mixture Model (GMM) parametric classifiers are supported. In those classifiers each class is represented respectively as a single multidimensional Gaussian distribution (MAP) or a mixture of multidimensional Gaussian distributions (GMM). As an example of non-parametric classifier we support the K nearest neighbors (KNN) family of classifiers. In those classifiers each sample is classified according to the class of the majority of its K nearest samples in the training set. For clustering the well known *c-means* algorithm is used. The same algorithm is also used for *vector quantization*, a technique where each feature vector is coded as an integer index to a codebook of representative feature vectors corresponding to the means of the clusters. An intuitive way to think about classification algorithms is that they try to partition the high dimensional feature space into regions so that the vectors falling in one region come from the same class (see Figure 1).

For now we have implemented and evaluated two case studies of classification: a music/speech classifier and a genre classifier. The music/speech classifier is based on work described in [11] and achieves 90.1% classification accuracy. The genre classifier uses three classes/genres to describe the data: classical, modern (rock, pop) and jazz. It achieves classification accuracy of 75% which is much better than chance. In many cases the errors make perceptual sense. For example, a jazz piece with singing and string accompaniment might be confused for a classical piece. No preprocessing of the data to avoid these outliers was done. The genre classification is done without taking into account any information about the beat of the song. It has been shown [10] that beat tracking can be performed robustly without transcription. Therefore the genre classification result can possibly be further improved by the use of a beat tracking algorithm.

In both case the classification decision is done for each frame separately therefore the results are representative of real world unknown data. Frames from the same sound file are never split into training set and testing set to avoid false high accuracy because of the inter-file frame coherence. Finally to robustly calculate the classification accuracy many different random partitions of the data to training and testing data are used and the results are averaged. The dataset used for the development and evaluation of these classifiers consists of two hours of audio broken into 30 second sound files containing a variety of styles, textures, recording conditions and speakers.

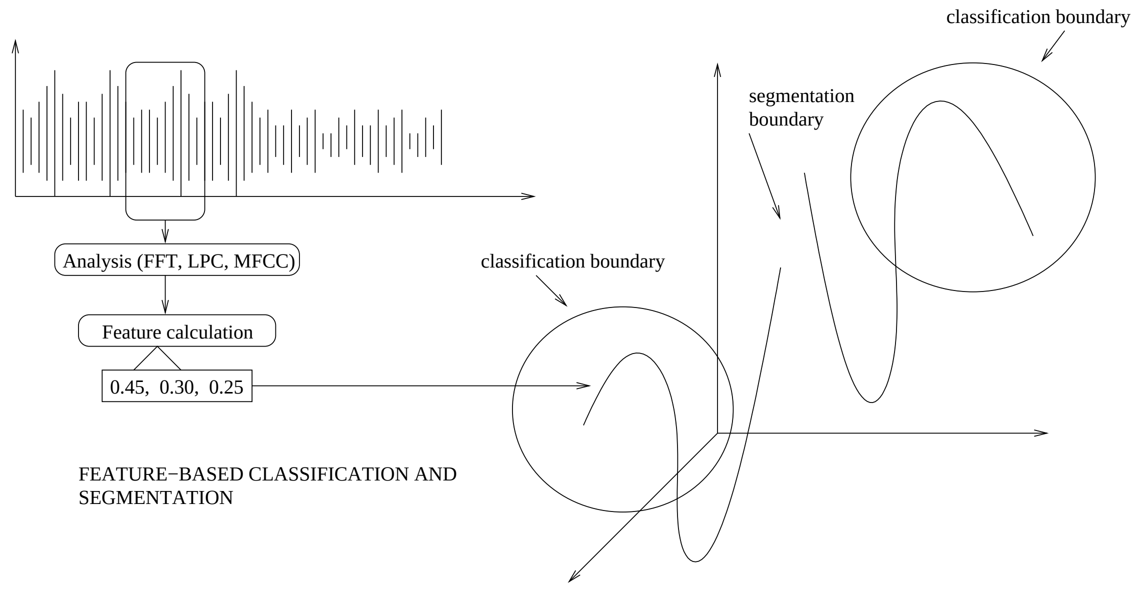


Figure 1: *Feature-based segmentation and classification*

3.2 Segmentation

Segmentation refers to the process of detecting when there is a change of "texture" in a sound stream. For example the chorus of a song, the entrance of a guitar solo, or the change from music to speech are all examples of segmentation boundaries. MARSYAS supports the general segmentation methodology described in [12] that uses tracking of multiple features in time. Intuitively the signal is viewed as a trajectory of points, corresponding to the feature vectors, in a high-dimensional space. The segmentation boundaries are found by detecting abrupt changes in this trajectory. User experiments confirming the validity of the method have been conducted in [13].

Segmentation is important for AIR because typically music sound files contain regions with different characteristics. Therefore it is necessary to treat the sound file as a collection of regions rather as one region characterized by its statistics. For example mixing the statistics of singing and a guitar solo will create problems for an AIR system.

3.3 Retrieval

In content-based AIR the query is a sound file and the result is a list of sound files ranked by their similarity. Three approaches are used in MARSYAS to represent a sound file for retrieval. In the first approach the sound file is represented as a single feature vector for the whole file. Ranking is performed by calculating a distance-metric between the vectors. This approach works well for small sound files that have the same "texture" throughout the piece.

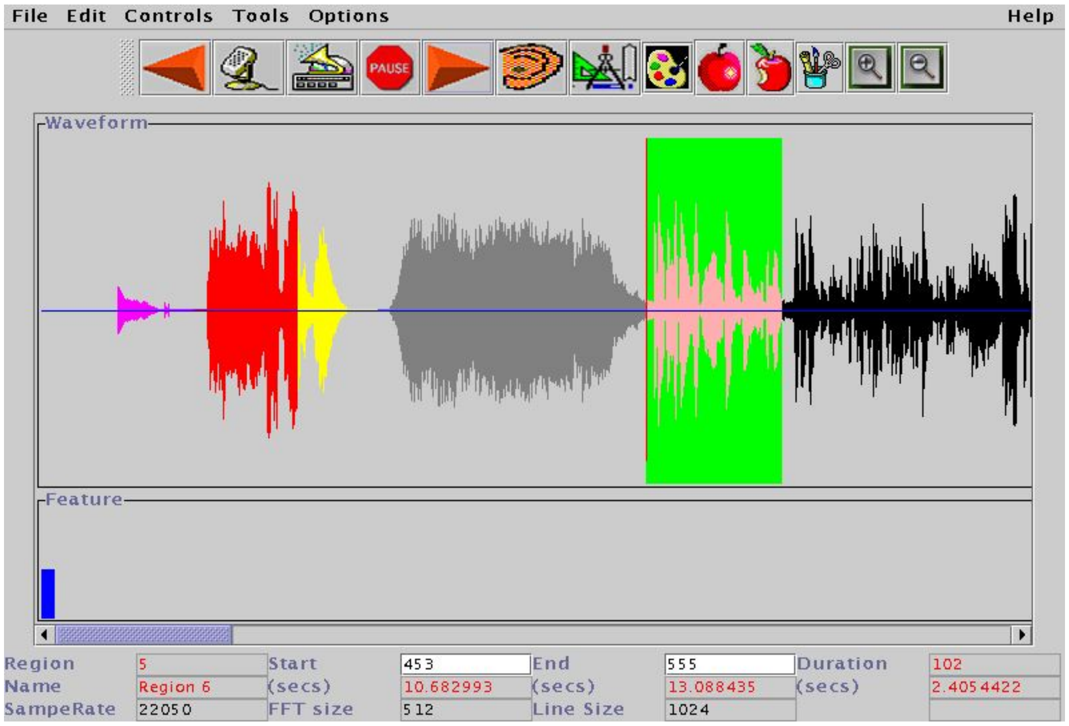
In the second approach the sound file is initially segmented using the algorithm described above and then each segment is characterized by a feature vector. Therefore each sound file is a variable length list of feature vectors. We define the distance of a segment A_i from a list of segments B to be the minimum segment to segment distance of A_i to each B_j. The distance of a sound file (a list A) is defined as the sum of those minimum distances for each A_i. This scheme works well with sound files that have more than one textures. It is time invariant in the sense that if we swap two segments in a sound file that will not change the distance it has from other files. Whether this feature is desired depends on the specific requirements of the application.

The third approach is to represent the sound file as a trajectory of feature vectors. There is no clear way to define a distance metric in this case. We have used the distance between the histograms of the feature vectors where each histogram bin is calculated using *c-means* clustering over a large dataset of representative sound files.

The best results have been obtained using the second approach. This observation is based on informal experiments. Content-based retrieval is subjective and the only way to properly evaluate a system is through user studies which we are planning to conduct in the future.

In order to perform user studies, a Web infastructure for evaluating AIR techniques was developed. As an initial test for the infastructure a user survey of AIR techniques was conducted using 1000 30-second long segments of classic rock songs. The large size of the dataset made the calculation of recall difficult so only precision was examined. Because the 30-second snippets used for the evaluation did not have many texture changes the single vector approach was used. The evaluation was done by 7 subjects who gave a relevance judgement from 1 (worse) to 5 (best). There were twelve queries, five matches returned and three algorithms (random, only beat-detection, beat and texture) giving a total of $7 * 5 * 12 * 3 = 1260$ data points. The beat detection was done using the algorithm described in [10]. The full retrieval was done using the beat detection and spectral features describing the texture of the song.

The random retrieval received a mean score of 2.1 (1.1 standard deviation), the beat-only mean was 2.9 (st.d 1.1) and the full algorithm mean was 3.1 (st.d 1.2). The standard deviation is due to the different nature of the queries and subject differences and is about the same for all algorithms. Although it is clear that the system performs better than random and that the full approach is slightly better than using only beat detection more work needs to be done to improve the scores. The main purpose of the study was to evaluate the user evaluation Web infastructure and confirm that there is potential in automated AIR.



Region	5	Start	453	End	555	Duration	102
Name	Region 6	(secs)	10.682993	(secs)	13.088435	(secs)	2.4054422
SampeRate	22050	FFT size	512	Line Size	1024		

Figure 2: *MARSYAS augmented sound editor*

3.4 Audio Thumbnailing

Audio Thumbnailing refers to the process of creating a short summary sound file from a large sound file in such a way that the summary best captures the essential elements of the original sound file. It is similar to the concept of key frames in video and thumbnails in images. Audio Thumbnailing is important for AIR especially for the presentation of the returned ranked list since it allows the user to quickly hear the results and make his selection.

In [4], two methods of audio thumbnailing are explored. The first is based on clustering and the second is based on the use of Hidden Markov Models (HMMs). According to the user experiments described in [4] both methods perform better than random selection and clustering is the best method of the two. In addition to the clustering method a segmentation-based method is supported in MARSYAS. In this method short segments around the segmentation boundaries are concatenated to form the summary. User experiments to compare this method with the clustering-based method are planned for the future.

4 Augmented SoundEditor

The MARSYAS sound editor (figure 2) offers the same functionality as a traditional sound editor. Waveform and spectrogram displays, mouse selection, playback status bar and zooming are supported. In addition to these typical features, a sound file can automatically be segmented with each region displayed with a different color. For quick browsing the user can move by regions and each region can be annotated with text. In addition regions can easily be added or deleted. Different classification schemes can be applied to each segmented region or to arbitrary selections. A synchronous feature display can be used to display user selected features together with the waveform. Finally audio thumbnailing is also supported.

5 TimbreGrams

TimbreGrams are a new graphical representation of sound. The main idea is to use color perception and the pattern recognition capabilities of the human visual system to depict timbral and temporal information. A *TimbreGram* is a series of vertical color stripes where each stripe corresponds to a short time feature vector. Time is mapped from left to right. The mapping of the feature vectors to color is performed using Principal Component Analysis (PCA) described in the next section. Sound textures that are similar have similar colors. In addition time periodicity is shown in color. For example the ABA structure of a file with singing and instrumental parts will visible as ABA in color blocks. Figure 3 shows the *Timbregrams* of six sound files (each 30 seconds long). Three of them contain speech and three contain classical music.

Although color information is lost in the greyscale of the paper, music and speech separate clearly. The bottom right sound file (opera) is light purple and the speech segments are light green. Light and bright colors typically correspond to speech or singing (figure 3 left column). Purple and blue colors typically correspond to classical music (figure 3 right column).

5.1 Principal Component Analysis

Principal Component Analysis (PCA) is a technique for dimensionality reduction [5]. Basically the extraction of a principal component ammounts to a variance maximizing rotation of the original variable space. In other words, the first principal component is the axis passing through the centroid of the feature vectors that has the maximum variance therefore explains a large part of the underlying feature structure. The next principal component tries to maximize the variance not explained by the first. In this manner, consecutive orthogonal components are extracted.

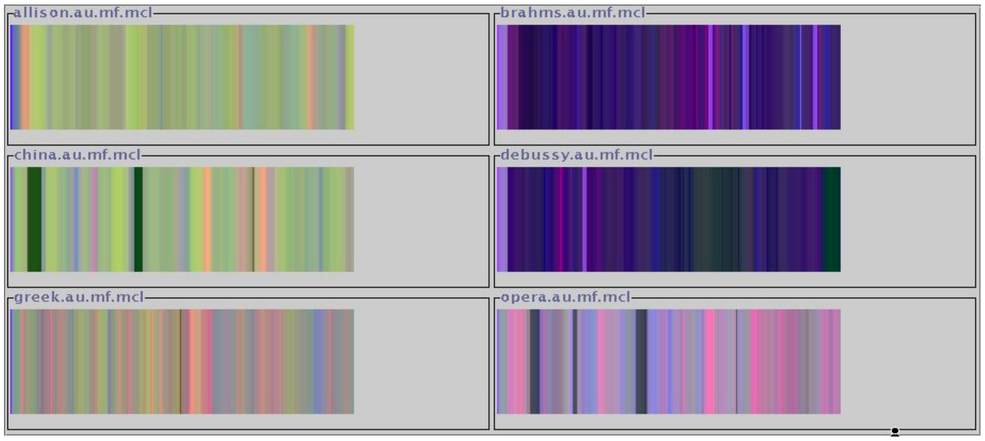


Figure 3: *Timbregrams for speech and classical music (RGB)*

The first three principal components of our feature vectors are mapped directly to color in either RGB or HSV color space. There seems to be a tradeoff regarding the color space used for the mapping. The RGB colorspace results in more uniformly colored *Timbregrams* that are aesthetically pleasing but blur the segmentation boundaries. Using the HSV colorspace shows better the segmentation boundaries with more contrasting stripes.

The three principal components used for the color mapping explain about 80% of the variance in our dataset. If only these three components are used for classification there is an improvement in classification accuracy. That implies that they characterize well the feature space and the remaining components are partly due to noise. We note that PCA is not the best method for feature subselection because it tries to characterize the whole dataset rather than trying to discriminate better the classes. Although *TimbreGrams* are purely data driven without any explicit class model it is easy to separate speech (light green), classical music (dark blue, purple) and rock (dark green) (with RGB mapping). In some cases color revealed outliers like a Philip Glass piece with electronic sounds and drums, originally labelled as classical, that has a dark green color suggesting rock or pop.

6 Implementation

All the tools described in this paper are integrated within MARSYAS an object-oriented framework for audio analysis [14]. The framework follows a client-server architecture. The server is a computation engine written in C++ that performs all the signal processing and pattern recognition and is optimized for real-time performance. The client written in JAVA contains the graphical user interface. MARSYAS has been tested on Linux, Solaris, IRIX and Windows (95,98,NT) systems.

7 Future work

In the future we plan to explore alternative visualizations for AIR. One obvious direction is the use of different color mapping schemes. The use of 2D and 3D space can provide additional dimensions for feature mapping.

Many of the techniques described in this paper work based on a training set of representative data. Having large and representative datasets is important in order to evaluate a system. Therefore, we are currently increasing the size of our datasets in some cases using automated tools. A more thorough hierarchical genre classification effort will be made with those larger datasets.

Finally in order to evaluate properly any AIR system extensive user experiments are required. User experiments for evaluating our retrieval and audio thumbnailing techniques are planned for the future. The use of a graphical user interface facilitated the experiments and data collection in [13]. Because our interface is written in JAVA it is easy to make it Web-based further facilitating the user experiments.

References

[1] R. Duda and P. Hart. *Pattern Classification and Scene Analysis.* John Wiley & Sons, 1973.

[2] J. Foote. An overview of audio information retrieval. *ACM Multimedia Systems*, 7:2–10, 1999.

[3] M. Hunt, M. Lennig, and P. Mermelstein. Experiments in syllable-based recognition of continuous speech. In *Proc.ICASSP*, 1980.

[4] B. Logan. Music summarization using key phrases. In *Proc.Int.Conf on Audio, Speech and Signal Processing, ICASSP*, 2000.

[5] L.T.Jolliffe. *Principal Component Analysis.* Springer-Verlag, 1986.

[6] J. Makhoul. Linear prediction: A tutorial overview. *Proc.IEEE*, 63:561–580, April 1975.

[7] K. Martin, E. Scheirer, and B. Vercoe. Musical content analysis through models of audition. In *Proc.ACM Multimedia Workshop on Content-Based Processing of Music*, Bristol, UK, 1998.

[8] D. Pye. Content-based methods for the management of digital music. In *Proc.Int.Conf on Audio, Speech and Signal Processing, ICASSP*, 2000.

[9] E. Scheirer. The mpeg-4 structured audio standard. In *Proc.Int.Conf on Audio, Speech and Signal Processing, ICASSP*, 1998.

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

[11] E. Scheirer and M. Slaney. Construction and evaluation of a robust multifeature speech/music discriminator. *IEEE Transactions on Acoustics, Speech and Signal Processing (ICASSP'97)*, pages 1331–1334, 1997.

[12] G. Tzanetakis and P. Cook. Multifeature audio segmentation for browsing and annotation. In *Proc.IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, WASPAA99*, New Paltz, NY, 1999.

[13] G. Tzanetakis and P. Cook. Experiments in computer-assisted annotation of audio. In *Proc. Int. Conf on Auditory Display, ICAD*, 2000.

[14] G. Tzanetakis and P. Cook. $MARSYAS$: A framework for audio analysis. *Organised Sound*, 2000. (to appear).

[15] G. Tzanetakis and P. Cook. Sound analysis using $MPEG$-compressed audio. In *Proc.Int.Conf on Audio, Speech and Signal Processing, ICASSP*, 2000.