A GENERIC CONTENT-BASED AUDIO INDEXING AND RETRIEVAL FRAMEWORK

Serkan Kiranyaz, Moncef Gabbouj

Institute of Signal Processing, Tampere University of Technology, Tampere, Finland
serkan@cs.tut.fi, moncef.gabbouj@tut.fi

ABSTRACT

Rapid increase in the amount of the digital audio collections presenting various formats, types, durations, and other parameters that the digital multimedia world refers, demands a generic framework for robust and efficient indexing and retrieval based on the aural content. Moreover, from the content-based multimedia retrieval point of view the audio information can be even more important than the visual part since it is mostly unique and significantly stable within the entire duration of the content. This paper presents a generic and robust audio based multimedia indexing and retrieval framework, which has been developed and tested under MUVIS system. This framework supports the dynamic integration of the audio feature extraction modules during the indexing and retrieval phases and therefore, provides a test-bed platform for developing robust and efficient aural feature extraction techniques. Furthermore the proposed framework is designed based on the high-level content classification and segmentation in order to improve the speed and accuracy of the aural retrievals. Both theoretical and experimental results are finally presented, including the comparative measures of retrieval performance with respect to the visual counterpart.

1. INTRODUCTION

The recent hardware and software improvements in the computer world with the increasing Internet usage have caused a massive usage of digital multimedia in several types, formats and quality. This, however, brings the storage and management problems of the multimedia collections, and especially efficient content-based retrieval of any particular media item becomes a challenge. In order to overcome such problems several content-based indexing and retrieval techniques and applications have been developed such as MUVIS system [14], [15], [16], [23], Photobook, VisualSeek, Virage, and VideoQ [28], [30], [33], [4]. The common feature of all such systems is that they all provide some kind of framework and several techniques for indexing and retrieving either still images or audio-video files. There are also several initiatives and standardization works such as MPEG-7 [10] for multimedia content description issues.

Yet the studies on content-based audio retrievals are still in the early stages. Traditional key-word based search engines such as Google, Yahoo, etc. usually cannot provide successful audio retrievals since they require costly (and usually manual) annotations that are obviously unpractical for large multimedia collections. In recent years, promising content-based audio retrieval techniques that might be categorized into two major paradigms have emerged. In the first paradigm, the "Query by Humming" (QBH) approach is tried for music retrievals. There are many studies in the literature, such as [1], [2], [5], [11], [18], [19], [21], [22]. However this approach has the disadvantage of being feasible only when the audio data is music stored in some symbolic format or polyphonic transcription (i.e. MIDI). Moreover it is not suitable for various music genres such as Trance, Hard-Rock, Techno and several others. Such a limited approach obviously cannot be a generic solution for the audio retrieval problem. The second paradigm is the well-known "Query by Example" (QBE) technique, which is also common for visual retrievals of the multimedia items. This is a more global approach, which is adopted by several research studies and implementations. One of the most popular systems is MuscleFish [24]. The designers, Wold et al. [32] proposed a fundamental approach to retrieve sound clips based on their content extracted using several acoustic features. In this approach, an N dimensional feature vector is built where each dimension is used to carry one of the acoustic features such as pitch, brightness, harmonicity, loudness, bandwidth, etc. and it is used for similarity search for a query sound. The main drawback of this approach is that it is a supervised algorithm that is only feasible to some limited sub-set of audio collection and hence cannot provide an adequate and global approach for general audio indexing. Furthermore, the sound clips must contain a unique content with a short duration. It does not address the retrieval problem in a generic case such as audio files carrying several and temporally mixed content along with longer and varying durations. Foote in [6] proposed an approach for the representation of an audio clip using a template, which characterizes the content. First, all the audio clips are converted in 16 KHz with 16 bits per sample representation. For template construction the audio clip is first divided into overlapped frames with a fixed duration and a 13-dimensional feature vector based on 12 mel frequency cepstrum coefficients (MFCC) and one spectral energy is formed for training a tree-based Vector Quantizer. For retrieval of a query audio clip, it is first converted into the template and then template matching is applied and ranked to generate the retrieval list. This is again a supervised method designed to work for short sound files with a single-content, fixed audio parameters and file format (i.e. au). It achieves an average retrieval precision within a long range from 30% to 75% for different audio classes. Li and Khokar [13] proposed a wavelet-based approach for the short sound file retrievals and presented a demo using the MuscleFish database. They achieved around 70% recall rate for diverse audio classes. Spevak and Favreau presented the SoundSpotter [31] prototype system for content-based audio section retrieval within an audio file. In their work the user selects a specific passage (section) within an audio clip and also sets the number of retrievals. The system then retrieves the similar passages within the same audio file by performing a pattern matching of the feature vectors and a ranking operation afterwards.

All the aforementioned systems and techniques achieved a certain performance; however present further limitations and drawbacks. First the limited amount of features extracted from the aural data often fails to represent the perceptual content of the audio data, which is usually subjective information. Second, the similarity matching in the query process is based on the computation of the (dis-) similarity distance between a query and each item in the database and a ranking operation afterwards. Therefore, especially for large databases it may turn out to be such a costly operation that the retrieval time becomes infeasible for a particular search engine or application. Third, all of the aforementioned techniques are designed to work in pre-fixed audio parameters (i.e. with a fixed format, sampling rate, bits per sample, etc.). Obviously, large-scale multimedia databases may contain digital audio that is in different formats (compressed or uncompressed), encoding schemes (MPEG Layer-2 [7], [26], MP3 [3], [7], [9], [26], AAC [3], [8], ADPCM, etc.), other capturing, encoding and acoustic parameters (i.e. sampling frequency, bits per sample, sound volume level, bit-rate, etc.) and durations. It is a fact that the aural content is totally independent from such parameters. For example, the same *speech* content can be represented by an audio signal sampled at 16 KHz or 44.1 KHz, in stereo or mono, compressed by MP3 in 64 Kb/s, or by AAC 24 Kb/s, or simply in (uncompressed) PCM format, lasting 15 seconds or 10 minutes, etc. However, if not designed accordingly, the feature extraction techniques are often affected drastically by such parameters and therefore, the efficiency and the accuracy of the indexing and retrieval operations will both be degraded as a result. Finally, they are mostly designed either for short sound files bearing a unique content or manually selected (short) sections. However, in a multimedia database, each clip can contain multiple content types, which are temporally (and also spatially) mixed with indefinite durations. Even the same content type (i.e. *speech* or *music*) may be produced by different sources (people, instruments, etc.) and should therefore, be analyzed accordingly.

In order to overcome the aforementioned problems and shortcomings, in this paper we propose a generic audio indexing and retrieval framework, which is developed and tested under the MUVIS system [14], [15], [23]. The primary objective in this framework is therefore, to provide a robust and adaptive basis, which performs audio indexing according to the audio class type (*speech*, *music*, etc.), audio content (the speaker, the subject, the environment, etc.) and the sound perception as closely modeled as possible to the human auditory perception mechanism. Furthermore, the proposed framework is designed in such a way that various low-level audio feature extraction methods can be used. For this purpose the internal **A**udio **F**eature **eX**traction (*AFeX*) framework can support various *AFeX* modules and it can also be used to develop new and efficient *AFeX* modules and then to test their efficiency against the conventional ones.

In order to achieve efficiency in terms of retrieval accuracy and speed, the proposed scheme uses high-level audio content information obtained from an efficient, robust and automatic (unsupervised) audio classification and segmentation algorithm [17] during both in indexing and retrieval processes. In this context, it is also optional for all *AFeX* modules so that a particular module can use the classification and segmentation information in order to tune and optimize its feature extraction process according to a particular class type. The audio classification and segmentation algorithm is especially designed for audio-based multimedia indexing and retrieval purposes. First of all, it has a multimodal structure, which supports both *bit-stream* mode for MP3 and AAC audio, and *generic* mode for any audio type and format within the MUVIS framework. In both modes, once a common spectral template is formed from the input audio source, the same analytical procedure is performed afterwards. The spectral template is obtained from MDCT coefficients of MP3 granules or AAC frames in *bit-stream* mode and hence called as MDCT template. The power spectrum obtained from FFT of the PCM samples within temporal frames forms the spectral template for the *generic* mode. Once the common spectral template is formed the granule/frame features can be extracted accordingly and thus, the primary classification and segmentation scheme can be built on a common basis, independent from the underlying audio format and the mode used. In order to improve the performance and most important of all, the overall accuracy, the classification scheme produces only 4 class types per audio segment: *speech, music, fuzzy* or *silent*. *Speech*, *music* and *silent* are the *pure* class types. The class type of a segment is defined as *fuzzy* if either it is not classifiable as a pure class due to some potential uncertainties or anomalies in the audio source or it exhibits features from more than one pure class. Therefore, for the proposed method, any erroneous classification on pure classes is intended to be detected as *fuzzy,* so as to avoid significant retrieval errors (mismatches) due to such potential misclassification. During the feature extraction operation, the feature vectors are extracted from each individual segment with a different class type and stored and retrieved separately. This makes more sense for content-based retrieval point of view since it brings the advantage to perform the similarity comparison between the frames within the segments with a matching class type and therefore, avoids any potential similarity mismatches and reduces the indexing and most important of all, the (query) retrieval times significantly.

In an audio retrieval operation the classification based indexing scheme is entirely used in order to achieve low-complexity and robust query results. The proposed *AFeX* framework supports merging of several audio feature sets and associated sub-features once the similarity distance per sub-feature is calculated. During the similarity distance calculation a penalization mechanism is developed in order to penalize not fully (partly) matched clips. For example if a clip with both *speech* and *music* parts is queried, all the clips missing one of the existing class type (say a music-only clip) will be penalized by the amount of the missing class (*speech*) coverage in the queried clip. This will give the priority to the clips with the entire class matching and therefore, ensure a more reliable retrieval. Another mechanism is applied for normalization due to the variations of the audio frame duration in the sub-features. This will change the number of frames within a class type and hence brings the dependency of the overall sub-feature similarity distance to the audio frame duration. Such dependency will negate any sub-feature merging attempts and therefore, normalization per audio per frame is applied. MUVIS framework internally provides a weighted merging scheme in order to achieve a "good" blend of the available audio features.

The rest of this paper is organized as follows: in section 2, we outline an overview of MUVIS system where the proposed framework is embedded. Section 3 presents the proposed audio indexing and retrieval framework. We demonstrate several experimental results on aural multimedia retrieval via query in section 4 and finally section 5 concludes the paper.

2. THE MUVIS SYSTEM

2.1. System Overview

As shown in Figure 1, the MUVIS system is based upon three applications, each of which has different responsibilities and facilities. *AVDatabase* is mainly responsible for real-time audio/video database creation with which audio/video clips are captured, (possibly) encoded and recorded in real-time from any peripheral audio and video devices connected to a computer. *DbsEditor* performs the visual and aural indexing of the multimedia databases and therefore, offline feature extraction process over the multimedia collections is its main task. *MBrowser* is the primary media browser and retrieval application into which an enhanced query technique, the *Progressive Query* (*PQ*) [16], is integrated as the primary retrieval (QBE) scheme. The traditional query technique, the *Normal Query* (*NQ*) [16], is the alternative query scheme within *MBrowser*. A Metric Access Method (MAM) based indexing scheme, *Hierarchical Cellular Tree* (*HCT*) [23], has been recently integrated into MUVIS. Both *PQ* types (Sequential *PQ* and *PQ* over *HCT*) can be used for retrieval of the multimedia primitives with respect to their similarity to a queried media item (an audio/video clip, a video frame or an image). During a query operation the similarity distances between the query and the database items will be calculated by the particular functions, each of which is implemented in the corresponding visual/aural feature extraction (*FeX*/*AFeX*) modules. More detailed information about MUVIS can be found in [23].

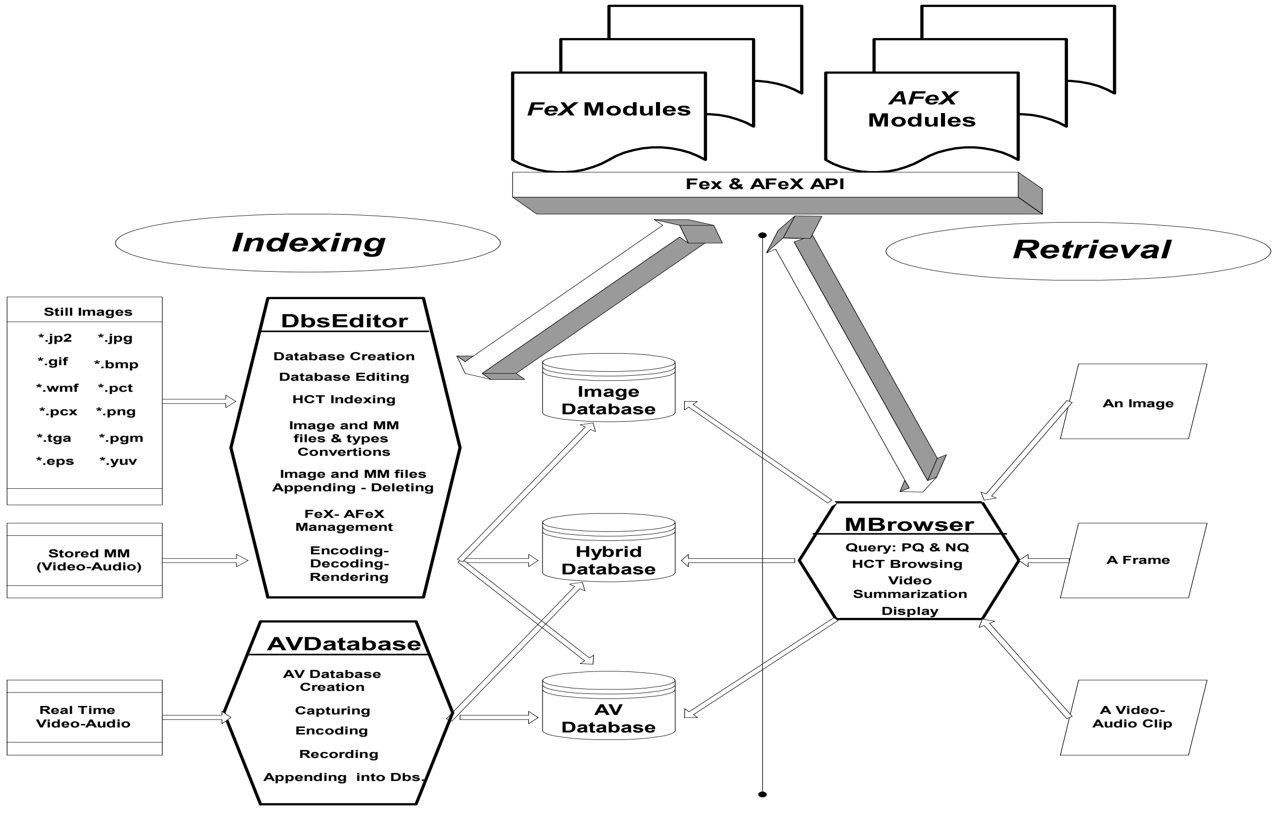


Figure 1: General structure of MUVIS framework

Table I: MUVIS Multimedia Family.

	MUVIS Audio					MUVIS Video		
Codecs	**Sampling Freq.**	**Channel No**	**File Formats**		**Codecs**	**Frame Rate**	**Frame Size**	**File Formats**
MP3	16, 22.050,	Mono	MP3		H263+	1..25 fps	Any	AVI
AAC	24, 32, 44.1 KHz	Stereo	AAC		MPEG-4			MP4
G721	Any		AVI		YUV 4:2:0			
G723								
PCM	Any		MP4		RGB 24			

The MUVIS system supports the following types of multimedia databases:

- Audio/Video databases include only audio/video clips and associated indexing information.
- Image databases include only still images and associated indexing information.
- Hybrid databases include both audio/video clips and images, and associated indexing information.

MUVIS databases can only contain the multimedia types belonging to MUVIS multimedia family as given in Table I. Alien formats (e.g. MPEG-1 video and audio) can be converted by *DbsEditor* to one of the supported formats first and then appended. Audio (only) files can be captured, encoded, recorded and appended in real-time similar to the video via *AVDatabase* or can be appended by conversion via *DbsEditor*. For audio encoding, the last generation audio encoders achieving high aural quality even in very low bit-rates such as MP3 and AAC can be used. ADPCM encoders such as G721 and G723 can also be used for their low complexity. Furthermore, audio can be recorded in uncompressed (raw PCM 16b/s) format. Compressed audio bit-stream is then recorded into any audio-only file format (container) such as MP3 and AAC or possibly interlaced with video, such as AVI and MP4.

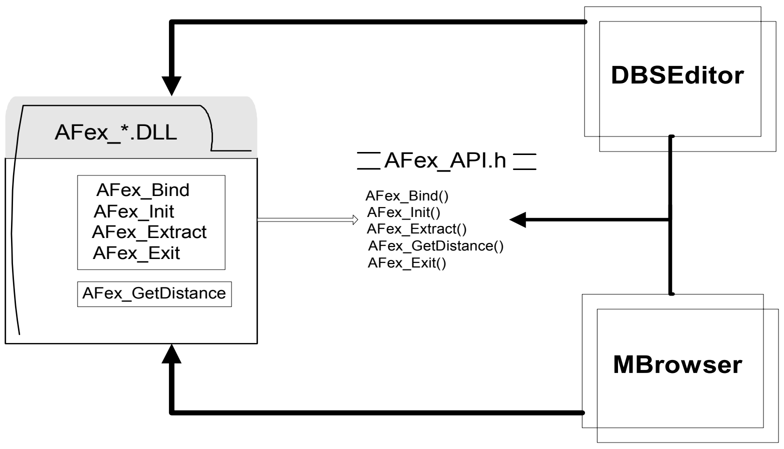


Figure 2: Basic *AFeX* Module interaction with MUVIS applications.

2.2. Aural Feature Extraction Framework: *AFeX*

AFeX framework mainly supports dynamic audio feature extraction module integration for audio clips. Figure 2 shows the API functions and linkage between MUVIS applications and a sample *AFeX* module. All audio feature extraction algorithms should be implemented as a Dynamically Linked Library (*DLL)* with respect to *AFeX* API. *AFeX* API provides the necessary handshaking and information flow between a MUVIS application and an *AFeX* module.

The details about audio indexing and a sample *AFeX* module will be explained in the next section.

3. AUDIO INDEXING AND RETRIEVAL FRAMEWORK IN MUVIS

Audio is an important source of information for content-based multimedia indexing and retrieval and it can sometimes be even more important then the visual part since it shows a stable behavior (i.e. less variations and no abrupt changes) according to the content. However, when dealing with digital audio there are several requirements to be fulfilled and the most important of them is the fact that the content is totally independent from the digital audio capture parameters (i.e. sound volume, sampling frequency, etc.), audio file type (i.e. AVI, MP3, etc.), encoder type (MP3, AAC, etc.), encoding parameters (i.e. bit-rate, etc.) and other variations such as duration and sound volume level. So the overall structure of audio-based indexing and retrieval framework is designed to provide a pre-emptive robustness (independency) to such parameters and variations.

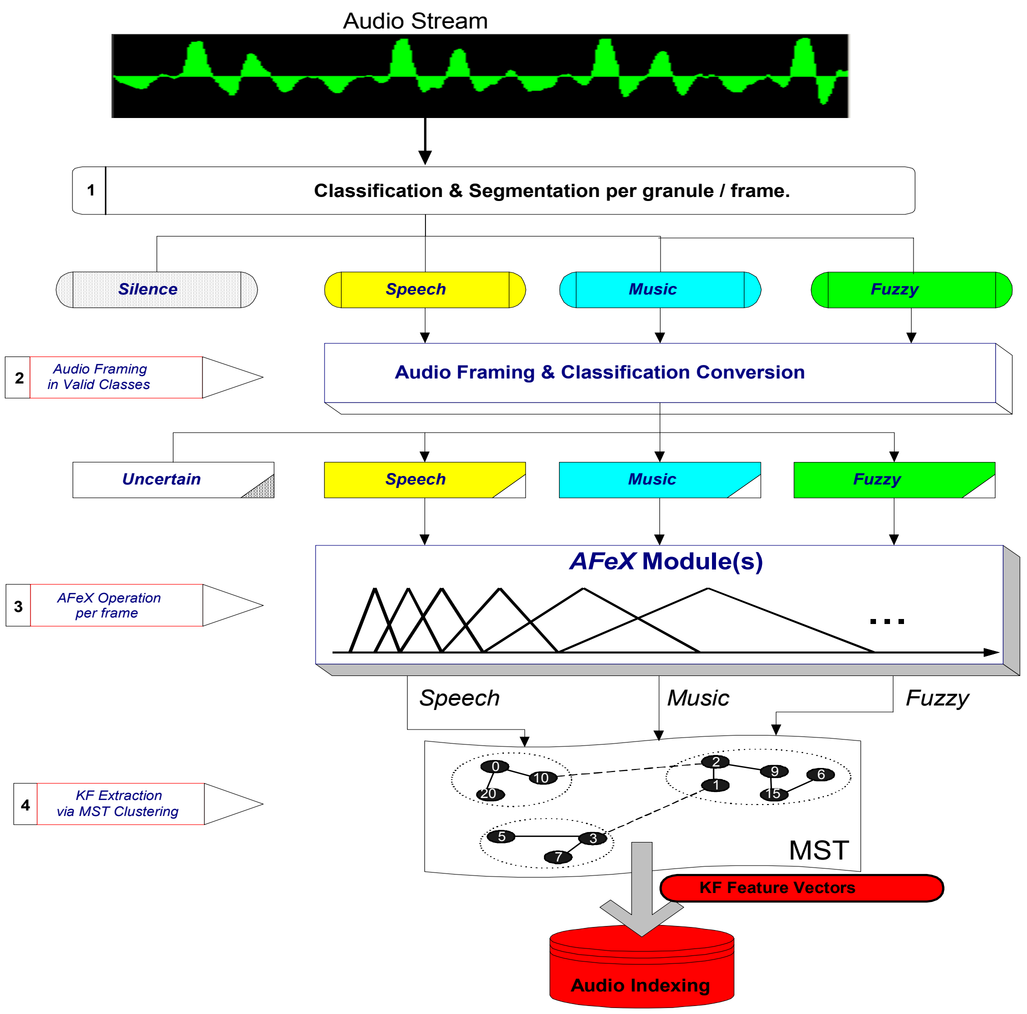


Figure 3: MUVIS Audio Indexing Operation Flowchart.

As shown in Figure 3, audio indexing is applied to each multimedia item in a MUVIS database containing audio, and it is accomplished in several steps. The classification and segmentation of the audio stream is the first step. As a result of this step the entire audio clip is segmented into 4 class types and the audio frames among three class types (*speech*, *music* and *fuzzy*) are used for indexing. *Silent* frames are simply discarded since they do not carry any audio content information. The frame conversion is applied in step 2 due to the (possible) difference occurred in frame durations used in classification and segmentation and the latter *AFeX* operations. The boundary frames, which contain more than one class types are assigned as *uncertain* and also discarded from indexing since their content is not pure, rather mixed and hence do not provide a clean content information. The remaining *speech*, *music* and *fuzzy* frames (within their corresponding segments) are each subjected to audio feature extraction (*AFeX*) modules and their corresponding feature vectors are indexed into descriptor files separately after a clustering (key-framing) operation via Minimum Spanning Tree (MST) Clustering [12]. In the following sub-sections we will detail each of the indexing steps.

3.1. Audio Classification and Segmentation Algorithm

In order to achieve suitable content analysis, the first step is to perform accurate classification and segmentation over the entire audio clip. The developed algorithm [17] is a generic audio classification and segmentation especially suitable for audio-based multimedia indexing and retrieval systems. It has a multimodal structure, which supports both *bit-stream* mode for MP3 and AAC audio, and a *generic* mode for any audio type and format. In both modes, once a common spectral template is formed from the input audio source, the same analytical procedure can be performed afterwards. It is also automatic (unsupervised) in a way that no training or feedback (from the video part or human interference) is required. It further provides robust (invariant) solution for the digital audio files with various capturing/encoding parameters and modes. In order to achieve a certain robustness level, a *fuzzy* approach has been integrated within the technique.

Furthermore, in order to improve the performance and most important of all, the overall accuracy, the classification scheme produces only 4 class types per audio segment: *speech, music, fuzzy* or *silent*. *Speech*, *music* and *silent* are the *pure* class types. The class type of a segment is defined as *fuzzy* if either it is not classifiable as a pure class due to some potential uncertainties or anomalies in the audio source or it exhibits features from more than one pure class. The primary use of such classification and segmentation scheme is the following: For audio based indexing and retrieval, a pure class content is only searched throughout the associated segments of the audio items in the database having the same (matching) pure class type, such as *speech* or *music.* All *silent* segments and *silent* frames within *non-silent* segments can be discarded from the audio indexing. Special care is taken for the *fuzzy* content, that is, during the retrieval phase, the *fuzzy* content is compared with all relevant content types of the database (i.e. *speech*, *music* and *fuzzy)* since it might, by definition, contain a mixture of pure class types, background noise, aural effects, etc. Therefore, for the proposed method, any erroneous classification on pure classes is intended to be detected as *fuzzy,* so as to avoid significant retrieval errors (mismatches) due to such potential misclassification. In this context, three prioritized error types of classification, illustrated in Figure 4 are defined:

- **Critical Errors:** These errors occur when one pure class is misclassified into another pure class. Such errors significantly degrade the overall performance of an indexing and retrieval scheme.
- **Semi-critical Errors:** These errors occur when a *fuzzy* class is misclassified as one of the pure class types. These errors moderately affect the performance of retrieval.
- **Non-critical Errors:** These errors occur when a pure class is misclassified as a *fuzzy* class. The effect of such errors on the overall indexing/retrieval scheme is negligible.

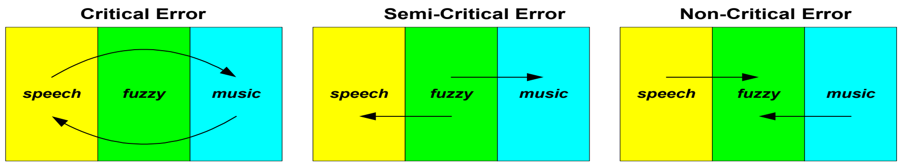
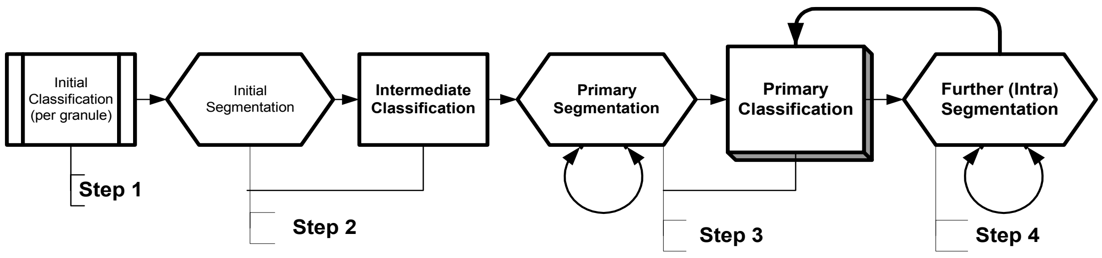


Figure 4: Different error types in classification.

The proposed approach is mainly developed based on the aforementioned fact: automatic audio segmentation and classification are mutually dependent problems. A good segmentation requires good classification and vice versa. Therefore, without any prior knowledge or supervising mechanism, the proposed algorithm proceeds in an iterative way, starting from granule/frame based classification and initial segmentation, the iterative steps are carried out until a global segmentation and thus a successful classification per segment can be achieved at the end. Figure 5 illustrates the 4-steps iterative approach to the audio classification and segmentation problem.


Initial Classification (per granule)
Initial Segmentation
Intermediate Classification
Primary Segmentation
Primary Classification
Further (Intra) Segmentation
Step 1
Step 2
Step 3
Step 4
Figure 5: The flowchart of the classification and segmentation algorithm.

The details about the classification and segmentation algorithm can be found in [17].

3.2. Audio Framing within Valid Audio Segments

As mentioned in the previous section, there are three valid audio segments: *speech, music* and *fuzzy*. Since segmentation and classification are performed per granule/frame basis, such as per MP3 granule or AAC frame, a conversion is needed to achieve a generic audio framing for indexing purposes. The entire audio clip is first divided into a user or model-defined audio frames, each of which will have a classification type as a result of the previous step. In order to assign a class type to an audio frame, all the granules/frames within or neighbor of that frame should have a unique class type to which it is assigned; otherwise, it will be assigned as *uncertain*.

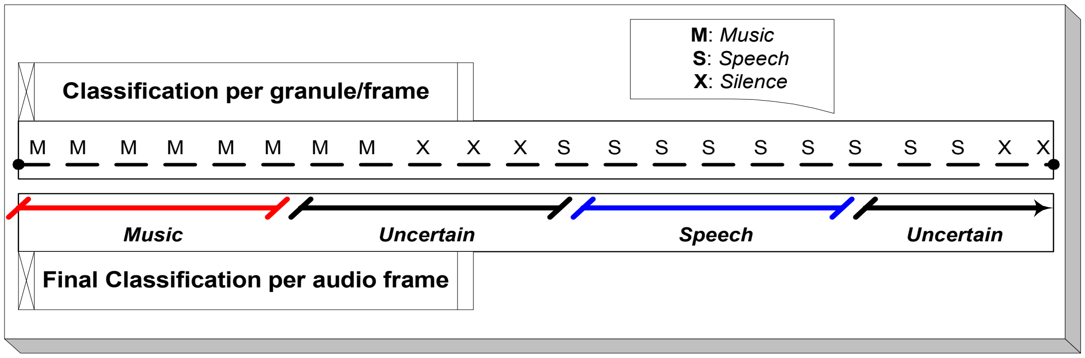

M: Music
S: Speech
X: Silence
Classification per granule/frame
M M M M M M M M X X X S S S S S S S S S X X
Music
Uncertain
Speech
Uncertain
Final Classification per audio frame

Figure 6: A sample audio classification conversion.

Since the *uncertain* frames are mixed and hence they are all transition frames (i.e. *music* to *speech*, *speech* to *silence*, etc.) the feature extraction will result an unclear feature vector, which does not contain a clean content characteristics at all. Therefore, these frames should be removed from the indexing operation thereafter.

3.3. A Sample *AFeX* Module Implementation: MFCC

MFCC stands for Mel-Frequency Cepstrum Coefficients [29] and they are widely used in several speech and speaker recognition systems due to the fact that they provide a decorrelated, perceptually-oriented observation vector in the cepstral domain and therefore, they are suitable for the human audio perception system. This is the main reason that we use them for audio based multimedia indexing and retrieval in order to achieve a similarity measure close to ordinary human audio perception criteria such as '*sounds like*' with additional higher level content discrimination via classification (i.e. *speech, music*, etc.).

The MFCC *AFeX* module performs several steps to extract MFCC per audio frame. First the incoming frames are Hamming windowed in order to enhance the harmonic nature of the vowels in *speech* and voiced consonants (sounds from instruments, effects, etc.) in *music*. In addition, Hamming window can reduce the effects of discontinuities and edges that are introduced during the framing process. Especially in logarithmic domain, the windowing effects can be encountered significantly. Hamming window is a half of cosine wave shifted upwards, as given in the following formula:

$$w(k) = 0.54 - 0.46\cos(2\pi \frac{k-1}{N-1}) \qquad (1)$$

where *N* is the size of the window, which is equal to the size (number of PCM samples) of the audio frames. In order to perform filtering in the time domain, the audio frame is zero-padded to get the size as a power of 2 and then FFT is applied to get into the spectral domain for plain multiplication with the filterbank. The mel (melody) scaled filterbank is a series of filterbank, which has the central frequencies uniformly distributed in mel-frequency (*mel(f)*) domain where

$$mel(f) = m_f = 1127\ln(1+\frac{f}{700}) \quad \text{and} \quad f = 700\,(e^{\frac{m_f}{1127}} - 1) \qquad (2)$$

Figure 7 illustrates a sample mel-scaled filterbank in the frequency domain. The number of bands is reduced for the sake of clarity. The shape of the band filters in the filterbank can be Hamming Window or plain triangular shape. As clearly seen in Figure 7 the resolution is high for low frequencies and low for higher frequencies. That is in tune with the human ear nature and one of the main reasons of the mel-scale usage. Once the filtering is applied, the energy is calculated per band and Cepstral Transform is applied on the band energy values. Cepstral Transform is a discrete cosine transform of log filterbank amplitudes:

$$c_i = (2/P)^{1/2} \sum_{j=1}^{P} \log m_j \cdot \cos\left(\frac{\pi \cdot i}{N}(j-0.5)\right) \qquad (3)$$

where $0 < i \le P$ and *P* is the number of filter banks. A subset of c_i is then used as the feature vector for this frame.

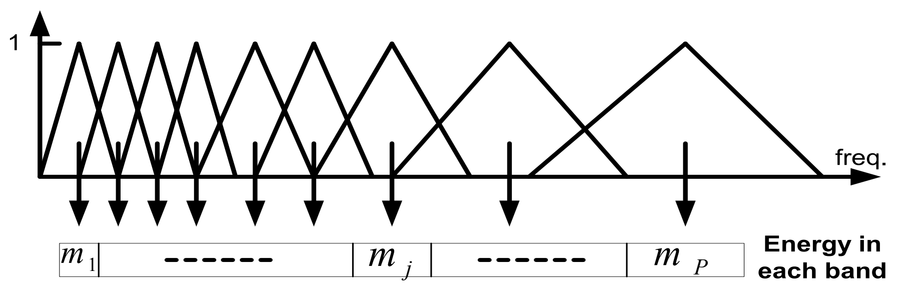


Figure 7: The derivation of mel-scaled filterbank amplitudes.

As mentioned in the previous sections, any *AFeX* module should provide generic feature vectors independent from the following variations:

- Sampling Frequency.
- Number of audio channels (mono/stereo).
- Sound Volume level.

By using audio data from only one channel for *AFeX* operation, the effect of multiple audio channels can be avoided. However, we need normalization during the calculation of the energy per filterbank in order to neutralize the effects of sampling frequency and volume variations. Let f_S be the sampling frequency. According to the *Nyquist* theorem, the bandwidth of the signal will be: $f_{BW} = f_s / 2$. The frequency resolution (Δf) per FFT spectral line will then be:

$$\Delta f = \frac{f_{BW}}{N_{FL}/2} = \frac{f_S}{N_{FL}} \qquad (4)$$

Let *T* be the duration (in milliseconds) of the incoming audio frames. Then the number of PCM samples within an audio frame will be: $N = T \cdot f_s / 1000$. An audio clip sampled with different sampling frequencies will result into different energy per band calculations due to the fact that the number of samples within the frame is varying and therefore, the band energy values should be normalized by a generic coefficient λ where $\lambda \sim N$.

Sound Volume (*V*) can be approximated as the absolute average level within the audio frame such as:

$$V \cong \frac{\sum_{i}^{N} |x_i|}{N} \qquad (5)$$

Similarly an audio clip with different volume levels will result into different energy per band calculations and therefore, the energy values should be normalized by λ where $\lambda \sim V$. The overall normalization will be:

$$\lambda \sim \lambda_V \cdot \lambda_f \sim V \cdot N \rightarrow \lambda \sim \sum_{i}^{N} |x_i| \qquad (6)$$

During the calculation of the band energies under each filterbank, the energy values are divided by λ to prevent both volume and sampling frequency effects over the Cepstrum coefficients calculation. As shown in Figure 7, the filterbank central frequencies are uniformly distributed over the mel-scale. Let f_{CF}^{i} be the center frequency of the i^{th} filter bank, then the filterbank central frequencies can be obtained by the following equation:

$$mel(f_{CF}^{i}) = \frac{i \cdot mel(f_{BW})}{P} \qquad (7)$$

So it is clear that the central frequencies will also be dependent on the sampling frequency ($f_{BW} = f_s / 2$). This brings the problem that the audio clips with different sampling frequencies will have filterbanks with different central frequencies and hence the feature vectors (MFCC) will be totally uncorrelated since they are derived directly from the band energy values from each filter bank. In order to fix the filterbank locations, we use a fixed cut-off frequency that corresponds to the maximum sampling frequency value used within MUVIS. The minimum and maximum sampling frequencies within the proposed audio indexing framework are 16 KHz and 44.1 KHz; therefore,

$$mel(f_{CF}^{i}) = \frac{i \cdot mel(f_{FCO})}{P}, where\ f_{FCO} \geq 22050 \qquad (8)$$

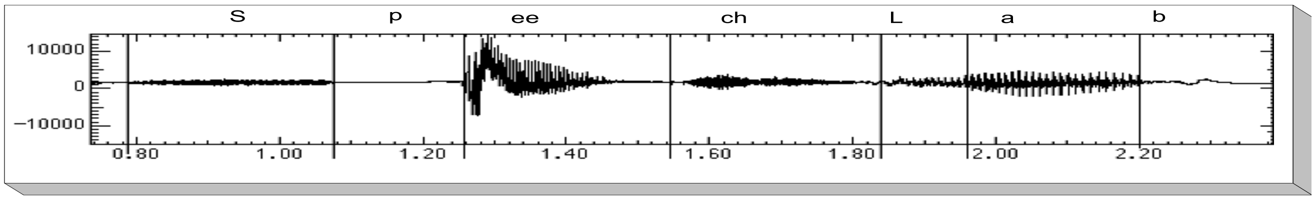
Setting the central frequencies by using the formula above will ensure the use of the same filterbank for all audio clips. Nevertheless, only the audio clips sampled at 44.1 KHz will use all the filters (assuming $f_{FCO} = 22050 Hz$) whilst the other audio clips sampled at lower frequencies will produce such band energy values of which the highest band values (m_j where *j>M*) are automatically set to 0 since those are outside of the bandwidth of the audio signal. This will yield erroneous results in the calculation of MFCC since the latter are nothing but DCT transforms of the logarithm of band energy values. In order to prevent this, only some portion of band energy values that are common for all possible sampling frequencies (within MUVIS) are used. In order to achieve this, the minimum possible *M* value is found using the lowest ($f_S = 16 KHz \Rightarrow f_{BW} = 8 KHz$) and highest ($f_S = 44.1 KHz \Rightarrow f_{BW} = 22.050 KHz$) sampling frequency values for MUVIS. Using $mel(8000) = 2840.03$ & $mel(22050) = 3923.35$ into Eq. 8, the bound for *M* can be stated as: $M \leq P \cdot 0.7238$. Therefore, having a filterbank that contains *P* band filters, we use *M* of them for the calculation of

MFCC. By this way only a common range of MFCC is therefore, used in order to negate the effect of the varying sampling frequencies of the audio clips within the database.

For indexing only the static values of Cepstral Transform coefficients (c_i) are used. The first coefficient is not used within the feature vector since it is a noisy calculation of the frame energy and hence it does not contain reliable information. The remaining M-1 coefficients over P ($c_i \forall 1 < i \leq M$) are used to form a MFCC feature vector.

3.4. Key-Framing via MST Clustering

The number of audio frames is proportional with the duration of the audio clip and once *AFeX* operation is performed, this may potentially result in a massive number of feature vectors, many of which are probably redundant due to the fact that the sounds within an audio clip are immensely repetitive and most of the time entirely alike. In order to achieve an efficient audio-based retrieval within an acceptable time, only the feature vectors of the frames from different sounds should be stored for indexing purposes. This is indeed a similar situation with the visual feature extraction scheme where only the visual feature vectors of the Key-Frames (KFs) are stored for indexing. There is however one difference: In the visual case KFs are known before the feature extraction phase but in aural case since there is no such physical 'frame' structure and hence the audio is framed uniformly with some certain duration, we need to obtain features of each frame beforehand in order to make Key-Frame analysis. This is why *AFeX* operation is performed (over valid frames) first and KFs are extracted afterwards.



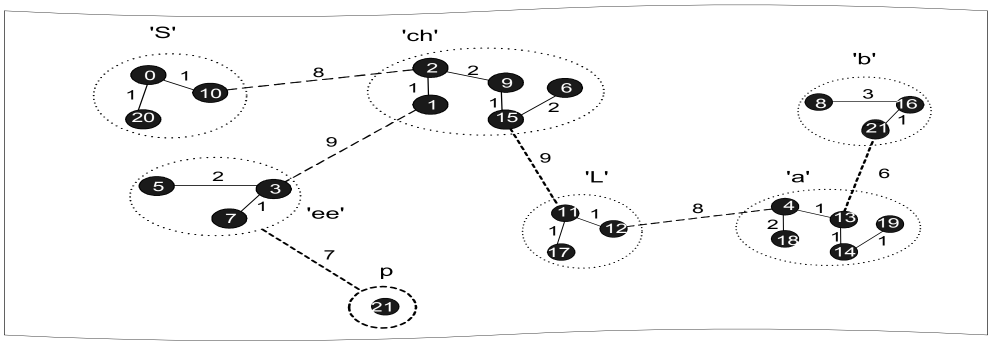


Figure 8: An illustrative clustering scheme.

In order to achieve an efficient KF extraction, the audio frames, which have similar sounds (and therefore, similar feature vectors) should first be clustered and one or more frame from each cluster should be chosen as a KF. An illustrative example is shown in Figure 8. Here the problem is to determine the number of clusters that should be extracted over a particular clip. This number will in fact vary with the content of the audio. For instance, a monolog *speech* will have less number of KFs than an action movie. For this we define *KF rate* that is the ratio between KF numbers over the total number of valid frames within a certain audio class type. Once a practical *KF rate* is set, the number of clusters can be easily calculated and eventually this number will be proportional to the duration of the clip. However, the longer clips will increase the chance of bearing similar sounds. Especially if the content is mostly based on *speech*, the similar sounds (vowels and unvoiced parts) will be repeated over time. Therefore, *KF rate* can be dynamically set via an empirical Key-Framing model that is shown in Figure 9.

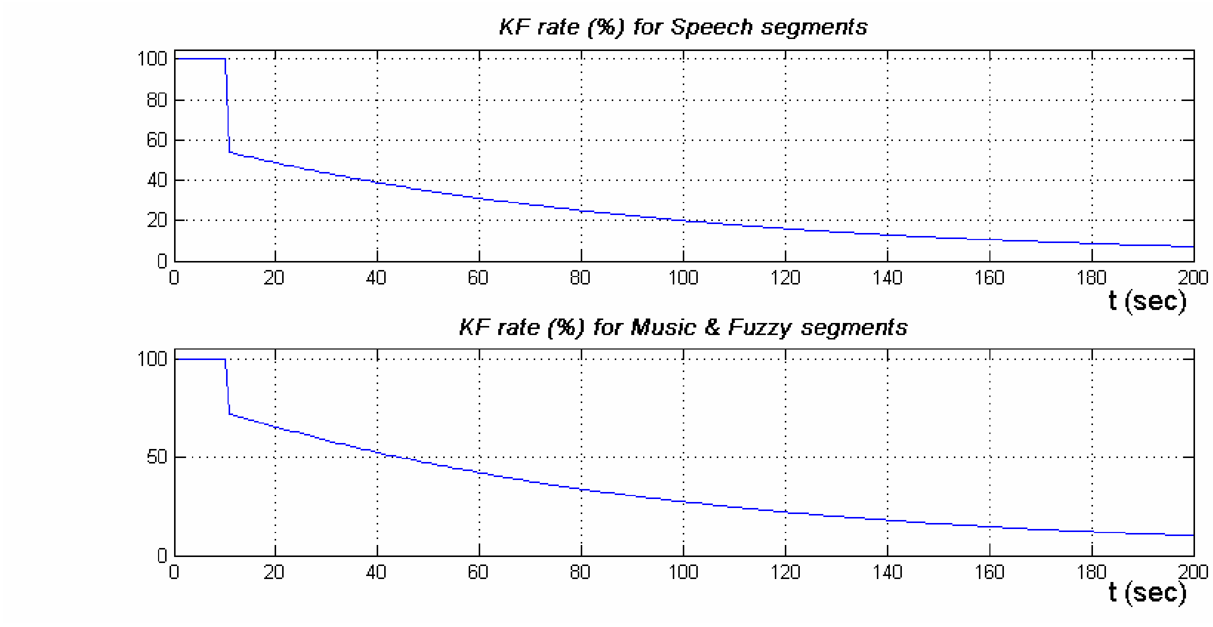


Figure 9: KF Rate (%) Plots.

Once the number of KFs (*KFno*) is set, the audio frames are then clustered using *Minimum Spanning Tree* (MST) clustering technique . Every node in MST is a feature vector of a unique audio frame and the distance between the nodes is calculated using the *AFeX* module *AFeX_GetDistance()* function. Once the MST is formed, then the longest *KFno*-1 branch is broken and as a result *KFno* clusters are obtained. By taking one (i.e. the first) frame as a KF, the feature vectors of the KFs are then used for indexing.

3.5. Aural Retrieval Scheme via Query-by-Example

As explained in detail in the previous sections, the audio part of any multimedia item within a MUVIS database is indexed using one or more *AFeX* modules that are dynamically linked to the MUVIS application. The indexing scheme uses the audio classification per segment information to improve the effectiveness in such a way that during an audio-based query scheme, the matching (same audio class types) audio frames will be compared with each other via the similarity measurement.

In order to accomplish an audio based query within MUVIS, an audio clip is chosen from a multimedia database and queried through the database if at least one or more audio features are extracted for the database. Let *NoS* be the number of feature sets existing in a database and let *NoF(s)* is the number of sub-features per feature set where $0 \le s < NoS$. As mentioned before sub-features are obtained by changing the *AFeX* module parameters or the audio frame size during the audio feature extraction process. Let the similarity distance function be $SD(x(s,f), y(s,f))$ where x and y are the associated feature vectors of the feature index s and the sub-feature index f. Let i be the index of the audio frames within the class C_q of the queried clip. Due to the aforementioned reasons, the similarity distance is only calculated between a sub-feature vector of this frame (i.e. $QFV_i^{C_q}(s,f)$) and an audio frame (index j) of the same class type from a clip (index *c)* within the database. For all the frames that have the same class type ($\forall j \Rightarrow j \in C_q$), one audio frame, which gives the minimum distance to the audio frame i in the queried clip is found ($D_i(s,f)$) and used for calculation of the total sub-feature similarity distance ($D(s,f)$) between two clips. Therefore, the particular frames and sections of the query audio are only compared with their corresponding (matching) frames and sections of a clip in database and this internal search will then provide the necessary retrieval robustness against the abrupt content variations within the audio clips and particularly their indefinite durations. Figure 10 illustrates the class matching and minimum distance search mechanisms during the similarity distance calculations per sub-feature. Furthermore, two factors should be applied during the calculation of $D(s,f)$ in order to achieve unbiased and robust results:

- **Penalization**: If no audio frames with class type C_q can be found in clip c then a penalization is applied during the calculation of $D(s,f)$. Let $N_Q(s,f)$ be the number of valid frames in queried clip and let $N_Q^{\varnothing}(s,f)$ be the number of frames that are not included for the calculation of the total sub-feature similarity distance due to the mismatches of their class types. Let $N_Q^{\Theta}(s,f)$ be the number of the rest of the frames, which will all be used in the calculation of the total sub-feature similarity distance. Therefore, $N_Q(s,f) = N_Q^{\varnothing}(s,f) + N_Q^{\Theta}(s,f)$ and the class mismatch penalization can be formulated as follows:

$$P_Q^C(s,f) = 1 + \frac{N_Q^{\varnothing}(s,f)}{N_Q(s,f)} \qquad (9)$$

If all the class types of the queried clip match with the class types of the database clip c, then $N_Q^{\varnothing}(s,f) = 0 \Rightarrow P_Q^C(s,f) = 1$ and this case naturally applies no penalization on the calculation of $D(s,f)$.

- **Normalization:** Due to the possibility of the variation of the audio frame duration for a sub-feature, the number of frames having a certain class types might change and this results in a biased (depending on the number of frames) similarity sub-feature distance calculation. In order to prevent this, $D(s,f)$ should be normalized by the total number of frames for each sub-feature ($N_Q(s,f)$). Therefore, this will yield a normalized $D(s,f)$ calculation, which is nothing but the sub-feature similarity distance per audio frame. Since the audio vectors are normalized the total query similarity distance (QD_c) between the queried clip and the clip *c* in the database is calculated with a weighted sum. The weights per sub-feature *f*, of a feature set *s*, $W(s,f)$ can be used for experimentation in order to find an optimum merging scheme for the audio features available in the database. The following equation formalizes the calculation of QD_c .

$$D_i(s,f) = \begin{Bmatrix} \min \left[SD\left(QFV_i^{C_q}(s,f),\ DFV_j^{C_q}(s,f) \right) \right]_{j \in C_i} & if\ \ j \in C_q \\ 0 & if\ \ j \notin C_q \end{Bmatrix}$$

$$D(s,f) = \frac{P_Q^C(s,f)}{N_Q(s,f)} \cdot \sum_q \sum_i^{i \in Cq} D_i(s,f)$$

$$QD_c = \sum_s^{NoS} \sum_f^{NoF(s)} W(s,f) \cdot D(s,f) \qquad (10)$$

Calculation of QD_c as in Eq. 10 is only valid if there is at least one matching class type between the queried clip and the database clip *c*. If no matching class types exist, then QD_c is assigned as ∞ and hence it will be placed among the least matching clips (to the end of the query retrieval queue). This is an expected case since two clips have nothing in common with respect to a high-level content analysis, i.e. their mismatching audio class types per segment.

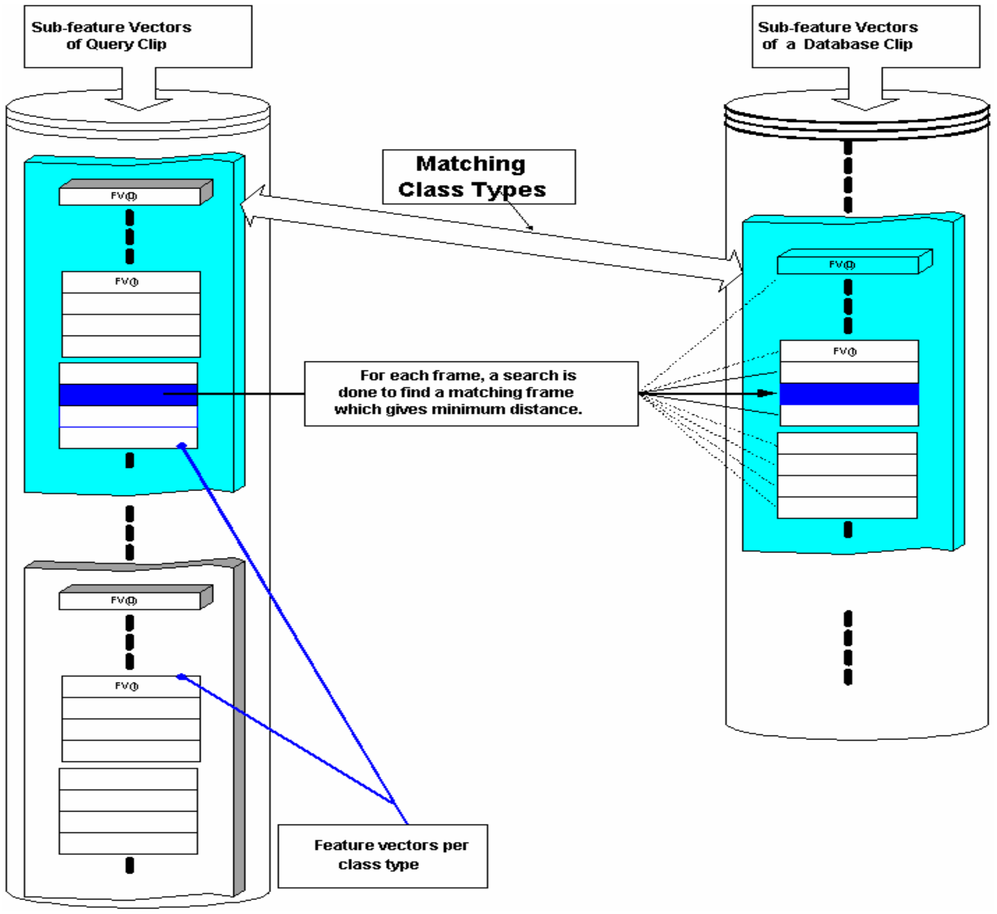

Sub-feature Vectors of Query Clip
Sub-feature Vectors of a Database Clip
Matching Class Types
FV(i)
For each frame, a search is done to find a matching frame which gives minimum distance.
Feature vectors per class type

Figure 10: A class-based audio query illustration showing the distance calculation per audio frame.

4. EXPERIMENTAL RESULTS

By means of MUVIS framework, we collected databases of multimedia clips offering various capturing and encoding parameters, practically an indefinite content supply and different formats and durations. Particularly, two different MUVIS databases are used in the experiments performed in this section:

1) ***Open*** Video Database: This database contains 1300 video clips, downloaded from "The Open Video Project" web site [25] and converted to a MUVIS database, which is available in [23]. The clips are quite old (from 1960s or older) but contain color video with sound. The total duration of the database is around 46 hours and the content mostly contains documentaries, talk shows, cartoons and commercials.

2) ***Real World*** Audio/Video Database: There are 800 audio-only and video clips in the database with a total duration of over 36 hours. They are captured from several TV channels and the content is distributed among news, commercials, talk shows, cartoons, sports and music clips.

All experiments are carried out on a Pentium-4 3.06 GHz computer with 2048 MB memory. All the sample MUVIS databases are indexed aurally using only MFCC *AFeX* module and visually using color (HSV and YUV color histograms), texture (Gabor [20] and GLCM [27]) and edge direction (Canny) *FeX* modules. The feature vectors are unit normalized and equal weights are used for merging sub-features from each of the *FeX* modules while calculating total (dis-) similarity distance. During the encoding and capturing phases, the acoustic parameters, codecs and the sound volume are kept varying among the potential values given in Table I. Furthermore, the clips in both databases have varying durations between 30 seconds to 3 minutes. The evaluation of the performance is carried out subjectively using only the samples containing clean content. In other words, if there is any subjective ambiguity on the result such as a significant doubt on the relevancy of any of the audio-based retrieval results from an aural (or a visual) query, etc., then that sample experimentation is simply discarded from the evaluation. Therefore, the experimental results presented in this section depend only on the decisive subjective evaluation via ground truth and yet they are meant to be evaluator-independent (i.e. same subjective decisions are guaranteed to be made by different evaluators).

For the analytical notion of performance along with the subjective evaluation, we used the traditional PR (Precision-Recall) performance metric measured under relevant (and unbiased) conditions, notably using the aforementioned ground truth methodology. Note that recall, *R*, and precision, *P*, are defined as:

$$R = \frac{RR}{TR} \quad and \quad P = \frac{RR}{N} \qquad (11)$$

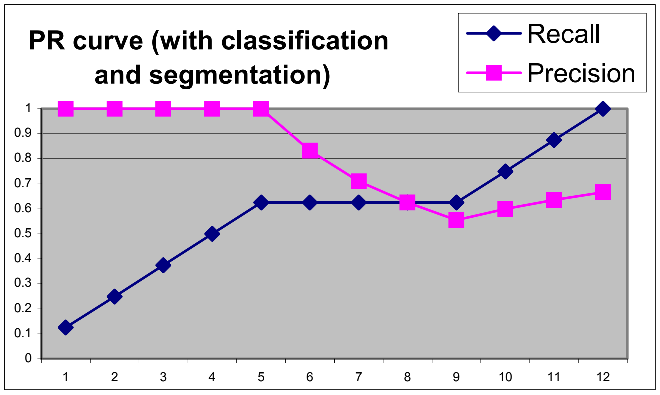
where *RR* is the number of relevant items retrieved (i.e. correct matches) among total number of relevant items, *TR*. *N* is the total number of items (relevant + irrelevant) retrieved. For practical considerations, we fixed *N* as 12. Recall is usually used in conjunction with the precision, which measures the fractional precision (accuracy) within retrieval and both can often be traded-off (i.e. one can achieve high precision versus low recall rate or vice versa.).

This section is organized as follows: First the effect of classification and segmentation (Step 1) over the total (indexing and) retrieval performance will be examined in the next sub-section. Afterwards, a more generic performance evaluation will be realized based on various aural retrieval experiments and especially the aural retrieval performance will be compared with the visual counterpart in an analytical and subjective (via visual inspection) way.

4.1. Classification and Segmentation Effect on Overall Performance

Several experiments are carried out in order to assess the performance effects of the audio classification and segmentation algorithm. The sample databases are indexed with and without the presence of audio classification and segmentation scheme, which is basically a matter of including/excluding Step-1 (the classification and segmentation module) from the indexing scheme. Extended experiments on audio based multimedia query retrievals using the audio classification and segmentation during the indexing and retrieval stages show that significant gain is achieved due to filtering the perceptually relevant audio content from a semantic point of view. The improvements in the retrieval process can be described based on each of the following factors:

- *Accuracy*: Since only multimedia clips, containing matching (same) audio content are to be compared with each other (i.e. *speech* with *speech*, *music* with *music*, etc.) during the query process, the probability of erroneous retrievals is reduced. The accuracy improvements are observed within 0-35% range for the average retrieval precision. One typical PR curve for an audio-based retrieval of a 2 minutes multimedia clip bearing pure *speech* content within **Real World** database is shown in Figure 11. Note that in the left part of Figure 11, 8 relevant clips are retrieved within 12 retrievals via classification and segmentation based retrieval. Without classification and segmentation, one relevant retrieval is clearly missed on the right side.



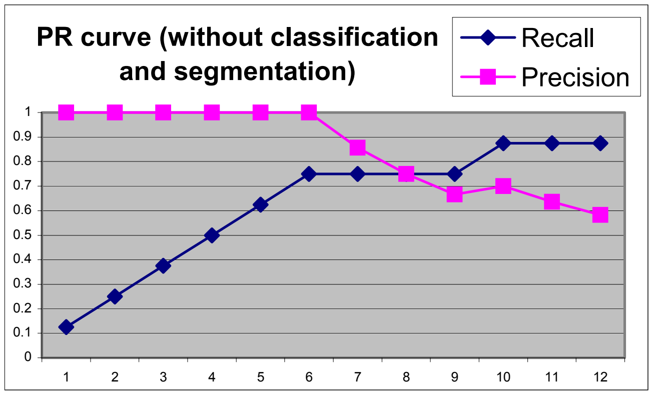


Figure 11: PR curves of an aural retrieval example within Real World database indexed with (left) and without (right) using classification and segmentation algorithm.

- *Speed*: The elimination of *silent* parts from the indexing scheme reduces the amount of data for indexing and retrieval and hence improves the overall retrieval speed. Moreover, the filtering of irrelevant (different) class types during the retrieval process significantly improves the speed by reducing the CPU time needed for similarity distance measurements and the sorting process afterwards. In order to verify this expectation experimentally and obtain a range for speed improvement, we have performed several aural queries on **Real World** database indexed with and without the application of classification and segmentation algorithm (i.e. Step-1 in indexing scheme). Among these retrievals we have chosen 10 of them, which have the same precision level in order to have an unbiased measure. Table 2 presents the total retrieval time (the time passed from the moment user initiates an aural query till the query is completed and results are displayed on the screen) for both cases. As a result the query speed improvements are observed within 7-60% range whilst having the same retrieval precision level.

Table 2: QTT (Query Total Time) in seconds of 10 aural retrieval examples from Real World database.

Aural Retrieval No.	1	2	3	4	5	6	7	8	9	10
QTT (*without classification and segmentation*)	47.437	28.282	42.453	42.703	43.844	42.687	46.782	45.814	44.406	41.5
QTT (*with classification and segmentation*)	30.078	26.266	19.64	39.141	18.016	16.671	31.312	30.578	20.006	37.39
QTT Reduction (%)	***36.59***	***7.12***	***53.73***	***8.34***	***58.9***	***60.94***	***33.06***	***33.25***	***54.94***	***9.90***

- *Disk Storage*: Fewer amounts of data are needed and henceforth recorded for the audio descriptors due to the same analogy given before. Furthermore the *silent* parts are totally discarded from the indexing structure. Yet it is difficult to give an exact analytical figure showing how much disk space can be saved by performing the classification and segmentation based indexing because this clearly depends on the content itself and particularly the amount of *silent* parts that the database items contain. The direct comparison between the audio descriptor file sizes of the same databases indexed with and without the proposed method shows that above 30% reduction can be obtained.

4.2. Experiments on Audio-Based Multimedia Indexing and Retrieval

For analytic evaluation, 10 aural and visual QBE (Query by Example) retrievals are performed according to the experimental conditions explained earlier. We only consider the first 12 retrievals (i.e. *N*=12) and both precision and recall values are given in Table 3 and 4.

Table 3: PR values of 10 Aural/Visual Retrieval (via QBE) Experiments in Open Video Database.

Query No:		1	2	3	4	5	6	7	8	9	10
Visual	*Precision*	0.66	0.75	0.25	0.25	0.66	1	0.58	1	0.83	1
	Recall	0.66	0.75	0.33	0.25	0.8	1	0.58	1	0.83	1
Aural	*Precision*	1	1	1	1	0.83	1	1	0.8	1	1
	Recall	1	1	1	1	1	1	1	0.8	1	1

Table 4: PR values of 10 Aural/Visual Retrieval (via QBE) Experiments in Real World Database.

Query No:		1	2	3	4	5	6	7	8	9	10
Visual	*Precision*	1	0.25	1	0.25	0.83	0.33	1	0.41	0.08	0.16
	Recall	1	0.5	1	0.25	1	0.33	1	0.41	0.125	0.66
Aural	*Precision*	1	0.5	1	0.5	0.83	0.75	1	0.75	0.25	0.25
	Recall	1	1	1	0.5	1	0.75	1	0.75	0.375	1

As the PR results clearly indicate in Table 3 and 4, in almost all the retrieval experiments performed, the aural queries achieved "equal or better" performance than their visual counterpart although there is only one feature (MFCC) used as the aural descriptor against a "good" blend of several visual features.

Figure 12 shows three sample retrievals via visual and aural queries from **Open** Video database using MUVIS *MBrowser* application. The query (the first key-frame in the clip) is shown on the top-left side of each figure. The first (top) example is a documentary about sharp-shooting competition and hunting in the USA. The audio is mostly *speech* with an occasional *music* and environmental noise (*fuzzy*). Among 12 retrievals, the visual query (left) retrieved three relevant clips (*P*=*R*=0.25) whereas the aural query retrieved all relevant ones (i.e. *P*=*R*=1.0). The second (middle) example is a cartoon with several characters and the aural content is mostly *speech* (dialogs between the cartoon characters) with long intervals of *music*. It can be easily seen that within 12 retrievals, the visual query (left) retrieved three relevant clips among 9 (*P*=0.25 and *R*=0.33) whereas the aural query retrieved all relevant ones within the first 9 ranks (i.e. *P*=*R*=1.0). Finally, the third example (bottom) is a commercial with an audio content that is *speech* with *music* in the background (*fuzzy*). Similarly, it is obvious that among 12 retrievals, the visual query retrieved 9 relevant clips (i.e. *P*=*R*=0.75) whereas the aural query retrieved all of them (i.e. *P*=*R*=1.0). All three examples show that the aural query can outperform the visual counterpart especially when there is a significant variation in the visual scenery, lightning conditions, background or object motions, camera effects, etc. whereas, the audio has the advantage of being usually stable and unique with respect to the content.

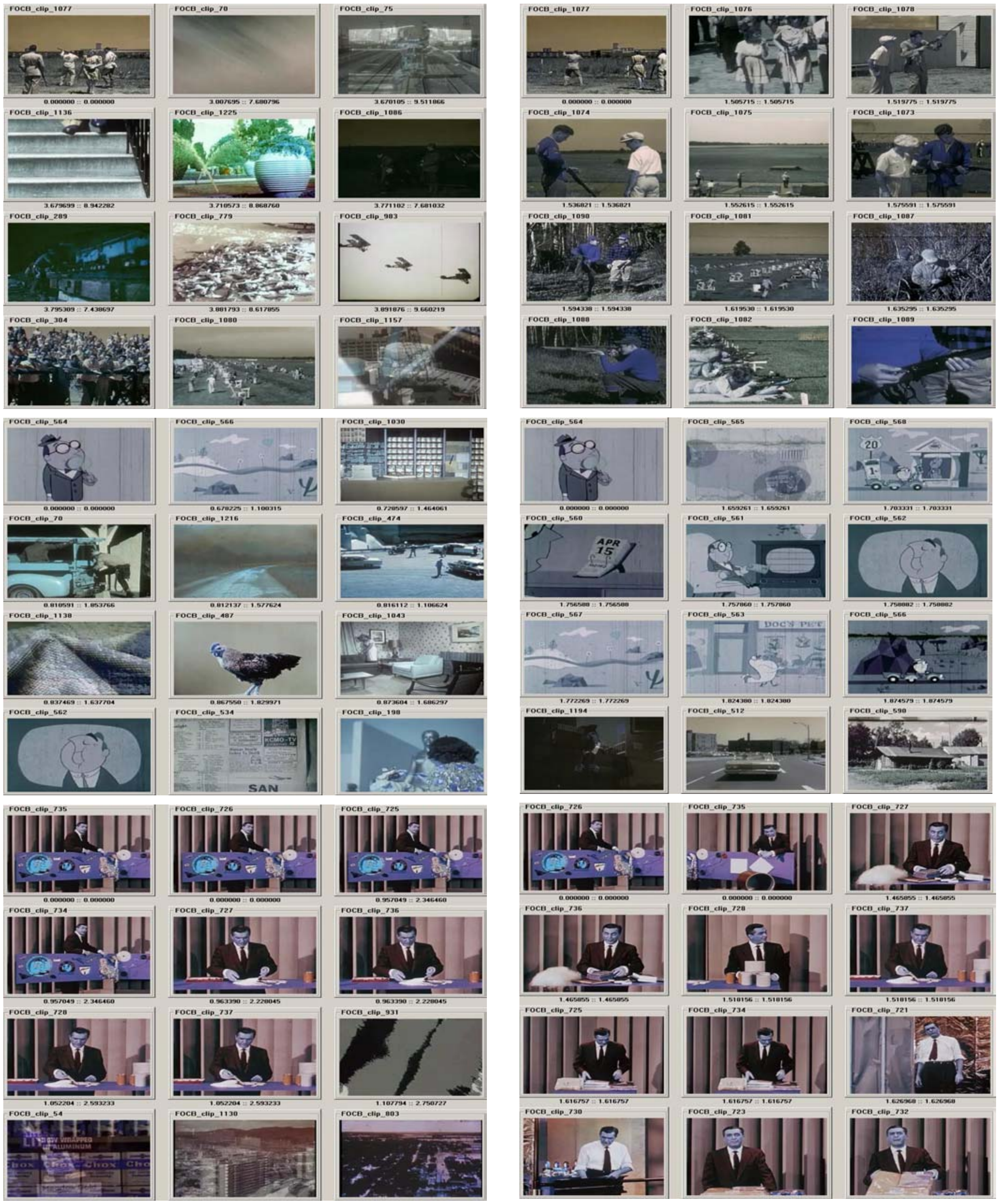


Figure 12: Three visual (left) and aural (right) retrievals in Open Video database. The top-left clip is the query.

5. CONCLUSIONS

In this paper a generic audio indexing and retrieval framework, which is integrated into the MUVIS system is proposed to achieve the following:

- An aural feature extraction modular structure has been developed in order to support dynamic integration of *AFeX* modules into the MUVIS applications. Such a modular approach, first of all, allows developing better aural feature extraction techniques, examining their efficiency and performance with respect to the traditional ones and combination of multiple *AFeX* modules in an efficient way in order to improve the retrieval performance and accuracy. Furthermore, this will provide a basis to support aural indexing of large multimedia databases since the existing *AFeX* modules can be tuned with respect to the content variations within the database and therefore, the retrieval efficiency in terms of speed and accuracy can further be optimized.
- One of the major advantages of the new audio indexing scheme is that it uses high-level audio content information during the feature extraction (optional) and indexing operations. Experimental results prove that such an approach significantly improves the indexing (in terms of storage, disk access and CPU computational time) and retrieval (in terms of speed and accuracy) performances.
- Further reduction in disk storage and increase in the speed of aural queries are achieved by introducing an empirical Key-Framing model which significantly reduces the massive volume of redundant (repetitive) frame features without causing a drastic degradation in the retrieval accuracy. By means of such adaptive approach, aural indexing and query of long multimedia clips becomes feasible.
- The overall infrastructure of the proposed framework is designed to provide a pre-emptive robustness (independency) to the broad range variations that a large multimedia database can offer within the capture, encoding and acoustic parameters of the digital audio present. Such an approach hence negates the side effects of such factors that are not related with the content information and hence provides a generic solution to the problem. Several retrieval experiments verify this fact by showing a loose dependency between the retrieval results and their aural parameters.
- Preliminary results show the effectiveness of the sample *AFeX* module introduced in this paper: MFCC. This particular *AFeX* module achieves a significant query performance, i.e. yielding “equal or better” retrieval accuracy as compared with the visual query, and furthermore it provides a robust and generic solution that is independent from the aforementioned factors.

Current and planned future studies include: the design of alternative and improved models for Key-Framing operation, the implementation of new *AFeX* modules purely based on the acoustic features of the audio information (i.e. pitch, bandwidth, etc.) and further optimization on the *AFeX* modules' discrimination power for an improved indexing and retrieval performance. Along with the static (frame-based) features, support for use of the dynamic features such as first and second MFCC derivatives is also considered.

6. REFERENCES

[1] D. Bainbridge, Extensible optical music recognition. PhD thesis, Department of Computer Science, University of Canterbury, New Zealand, 1997.

[2] S. Blackburn and D. DeRoure. "A Tool for Content Based Navigation of Music." In Proc. ACM Multimedia 98.

[3] K.-H. Brandenburg, "MP3 and AAC Explained*", AES 17th International Conference*, Florence, Italy, September 1999.

[4] S.F. Chang, W. Chen, J. Meng, H. Sundaram and D. Zhong, "VideoQ: An Automated Content Based Video Search System Using Visual Cues", *Proc. ACM Multimedia*, Seattle, 1997.

[5] T.-C. Chou , A. L. P. Chen , C.-C. Liu, "Music Databases: Indexing Techniques and Implementation", *Proc. of the 1996 International Workshop on Multi-Media Database Management Systems (IW-MMDBMS '96*), p.46, August 14-16, 1996.

[6] J. T. Foote, "Content-Based Retrieval of Music and Audio." *In Proc. SPIE, vol3229*, pp. 138-147, 1997.

[7] ISO/IEC 11172-3, Coding of Moving Pictures and Associated Audio for Digital Storage Media up to about 1.5 Mbit/s, Part 3: Audio, 1992.

[8] ISO/IEC CD 14496-3 Subpart4: 1998, Coding of Audiovisual Object Part 3: Audio, 1998.

[9] ISO/IEC 13818-3:1997, Information technology -- Generic coding of moving pictures and associated audio information -- Part 3: Audio, 1997.

[10] ISO/IEC JTC1/SC29/WG11, " Overview of the MPEG-7 Standard Version 5.0", March 2001.

[11] A. Ghias, J. Logan, and D. Chamberlin. B. C. Smith, "Query By Humming.", *In Proc. ACM Multimedia 95*, pp. 231-236, 1995.

[12] R.L. Graham and O. Hell, "On the history of the minimum spanning tree problem," *Annual Hist. Comput*. 7, pp. 43-57. 1985.

[13] A. Khokhar, G. Li "Content-based Indexing and Retrieval of Audio Data using Wavelet" *ICME* 2000.

[14] S. Kiranyaz, K. Caglar, O. Guldogan, and E. Karaoglu, "MUVIS: A Multimedia Browsing, Indexing and Retrieval Framework*", Proc. Third International Workshop on Content Based Multimedia Indexing, CBMI 2003*, Rennes, France, 22-24 September 2003.

[15] S. Kiranyaz, K. Caglar, E. Guldogan, O. Guldogan, and M. Gabbouj, "MUVIS: a content-based multimedia indexing and retrieval framework", *Proc. of the Seventh International Symposium on Signal Processing and its Applications, ISSPA 2003*, Paris, France, pp. 1-8, 1-4 July 2003.

[16] S. Kiranyaz, M. Gabbouj, "A NOVEL MULTIMEDIA RETRIEVAL TECHNIQUE: PROGRESSIVE QUERY (WHY WAIT?)", *5th Int. Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS 2004*, Lisboa, Portugal, 21-23 April, 2004.

[17] S. Kiranyaz, A. F. Qureshi and M. 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 2004*, pp. 55-62, London, UK, 25-26 November, 2004.

[18] L. Kjell and L. Pauli, "Musical Information Retrieval using musical Parameters", *International Computer Music Conference*, Ann Arbour, 1998.

[19] L. Lu, H. You, H. J. Zhang, "A New Approach to Query by Humming in Music Retrieval" in *Proc. of ICME 2001*, Tokyo,August 2001.

[20] W. Y. Ma, B. S. Manjunath, "A Comparison of Wavelet Transform Features for Texture Image Annotation", *Proc. IEEE International Conf. On Image Processing*, 1995.

[21] R.J. McNab, L.A.Smith, I.H. Witten, C.L. Henderson, and S.J. Cunningham, "Towards the digital music library: tune retrieval from acoustic input", *In Proceedings of ACM Digital Libraries* '96, 1118, 1996.

[22] R. J. McNab, L. A. Smith, D. Bainbridge, and I. H. Witten., “The New Zealand Digital Library MELody inDEX.” http://www.dlib.org/dlib/may97/meldex/05written.html, May 1997.
[23] MUVIS. http://muvis.cs.tut.fi/
[24] Muscle Fish LLC. http://www.musclefish.com/
[25] Open Video Project: http://www.open-video.org/
[26] D. Pan, “A tutorial on MPEG/Audio Compression”, *IEEE Multimedia*, pp 60-74, 1995.
[27] M. Partio, B. Cramariuc, M. Gabbouj, A. Visa, “Rock Texture Retrieval Using Gray Level Co-occurrence Matrix”, *Proc. of 5th Nordic Signal Processing Symposium*, Oct. 2002.
[28] A. Pentland, R.W. Picard, S. Sclaroff, “Photobook: tools for content based manipulation of image databases”, *Proc. SPIE (Storage and Retrieval for Image and Video Databases II)* 2185:34-37, 1994.
[29] L. R. Rabiner and B. H. Juang, Fundamental of Speech Recognition, Prentice hall, 1993.
[30] J.R. Smith and Chang, “VisualSEEk: a fully automated content-based image query system”, *ACM Multimedia*, Boston, Nov. 1996.
[31] C. Spevak and E. Favreau, “Soundspotter - a prototype system for content-based audio retrieval” *in Proc. of the COST G-5 Conf. on Digital Audio Effects* (DAFX-02), Hamburg, Germany, September 2002.
[32] E. Wold, T. Blum, D. Keislar, and J. Wheaton, "Content-based Classification, Search, and Retrieval of Audio", *IEEE Multimedia*, Fall 1996. pp. 27-36.
[33] Virage. URL:www.virage.com