

MUVIS: A CONTENT-BASED MULTIMEDIA INDEXING AND RETRIEVAL FRAMEWORK

Serkan Kiranyaz, Kerem Caglar , Esin Guldogan, Olcay Guldogan, Moncef Gabbouj*

Institute of Signal Processing, Tampere University of Technology, Tampere, Finland

* Nokia Mobile Software, Tampere, Finland

serkan@cs.tut.fi, kerem.caglar@nokia.com, moncef.gabbouj@tut.fi

ABSTRACT

There are primarily two MUVIS phases. In the first phase a Web-based application has been developed in late 90s to support indexing and retrieval in large image databases using visual and semantic features such as color, texture and shape. During recent years, a new framework, which aims to bring a unified and global approach on indexing, browsing and querying of various digital multimedia types such as audio, video and image has been developed. This new system provides tools for real-time audio and video capturing, encoding by last generation codecs such as MPEG-4, H.263+, MP3 and AAC. It supports several digital image types including JPEG 2000 and furthermore, it provides a well-defined interface for third parties to integrate their own feature extraction algorithms into the framework. In this paper, we describe the general system features with underlying applications and outline the main philosophy.

1. INTRODUCTION

In order to provide a solution for storage and management of massive digital multimedia data several content-based indexing and retrieval techniques and applications have been developed such as first and current MUVIS systems [1], [15], [16], Photobook, VisualSeek, Virage, and VideoQ [2],[3],[4],[5]. The primary objective of such systems is to provide a reliable basis and efficient techniques for indexing and retrieving digital image, audio and video information depending on the content that they bear. There are also several standard initiatives such as MPEG-7 [6] for multimedia content description issues.

In 1998, first MUVIS system has been implemented for indexing large image databases and retrieval via search and query techniques based on semantic and visual features [15]. Based upon the experience and feedback from this first system, recently a new PC-based MUVIS system, which is further capable of content-based indexing and retrieval of video and audio information in addition to several image types have been developed. The current MUVIS has been recently adopted by COST 211quat as the COST Framework

for CBIR. The current version of the MUVIS framework supports the following multimedia processing capabilities and features:

- Multimedia collection, conversion and creation of MUVIS databases,
- Real-time audio and video capturing, encoding and recording,
- Hierarchic video handling and representation,
- Content based indexing and retrieval of the several image formats such as JPEG, JPEG 2000 [20], BMP (Windows Bitmap), TGA, TIFF, PNG, PCX, PGM, GIF, WMF, PCT and EPS.
- Video summarization via scene detection [7],
- Handling of several multimedia file formats such as Microsoft AVI, MP4 (MPEG-4 v2 file format) [21], MP3 (MPEG-1 and MPEG-2 Layer 3) [10] and AAC (MPEG-2 and MPEG-4 Advanced Audio Codec) [11].
- Support for several audio and video encoding schemes such as H.263+, MPEG-4 (Simple Profile video codec), MP3, AAC, G721, G723. Uncompressed (raw) audio (PCM) and video (YUV 4:2:0 and RGB 24b) formats are also supported.
- An effective framework structure, which provides an application independent basis in order to develop audio and visual feature extraction techniques that are dynamically used for indexing and retrieval by the MUVIS applications.
- The retrieval based on distinct visual and aural queries initiated from a MUVIS database that includes audio/video clips.

The rest of this paper is organized as follows: in section 2 we outline the system philosophy of MUVIS and explain its functionalities and introduce the general MUVIS framework with underlying applications. Section 3 presents the visual and aural indexing framework with aural retrieval scheme. In section 4, we demonstrate some experimental results on querying and video summarization following with the conclusions, future remarks and discussions.

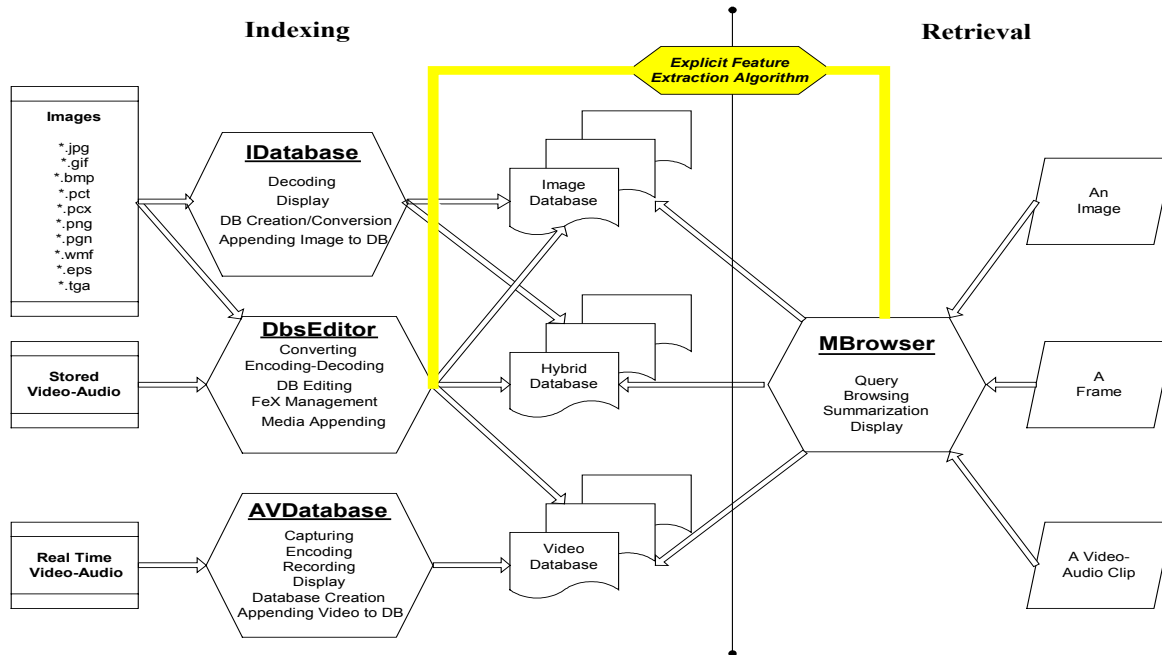


Figure 1: General structure of MUVIS framework

2. THE MUVIS SYSTEM

2.1. The System Overview

MUVIS aims to achieve a unified and global solution to the media (image, video and audio) capturing, recording, indexing and retrieval combined with browsing and various other visual and semantic capabilities.

As shown in Figure 1, MUVIS framework is based upon four applications, each of which has different responsibilities and facilities. *AVDatabase* and *IDatabase* applications are mainly responsible for video and image database creation, respectively. *DbsEditor* performs the indexing of the multimedia databases, and therefore, offline feature extraction processing is its main task. *MBrowser* is the primary media browser and retrieval application.

MUVIS system supports the following types of multimedia databases:

- *Video Databases* include only video clips and associated indexing information.
- *Image Databases* include only images and associated indexing information.
- *Hybrid Databases* include both video clips and images, and associated indexing information.

As illustrated in Figure 1, video databases can be created using *AVDatabase*, and image databases can be created using *IDatabase*. Hybrid databases can be created by appending images to video databases using *IDatabase* or *DbsEditor*, or by appending video clips to image databases using *DbsEditor* and *AVDatabase* applications.

2.2. Applications of MUVIS System

As briefly mentioned in the previous section, MUVIS backbone framework consists of 4 primary applications. In the following subsections we review the basic features of each application while emphasizing their role in the overall system.

2.2.1. AVDatabase

AVDatabase application is specifically designed and developed for creating Audio/Video Databases by indexing real-time audio/video while capturing it from a peripheral device. An audio/video clip may include only video information, only audio information or both video and audio information. In order to achieve optimal efficiency in recording and indexing, video and audio compression techniques are used by the application.

Video can be captured from any peripheral video source (i.e. PC camera, TV card, etc.) in one of the following formats:

- YV12 (I420) → YUV 4:2:0 (default)
- RGB24 or RGB32 (conversion required)
- YUY2 (YUYV) or UYVY (conversion required)

Capturing parameters such as capturing frame-rate, capturing image size can be defined in the application. If requested, an inbuilt H.263+ video encoder or an MPEG-4 (Simple Profile) encoder installed on the host system can be used. Video encoding parameters such as bit-rate, frame-rate, forced-intra rate (if enabled), etc. can be defined during the recording time. The supported file formats handle the synchronization of video with respect to the encoding timing of each frame.

Audio can be captured, encoded and recorded in real-time similar to the video. For audio encoding, we also use last generation audio encoders giving significantly high quality even in very low bit-rates such as *MP3* (MPEG-1 and MPEG-2 Layer 3) [10] and *AAC* (Advanced Audio Codec) [11]. *ADPCM* can also be used for low complexity. Audio can also be recorded in raw (*PCM*) format.

AVDatabase supports various file format options to store the audio and video information such as AVI, MP4 (*MPEG-4 v2 file format*), MP3 (*MPEG Layer-3*) and AAC (*MPEG-2/4 Advanced Audio Codec*) file formats.

2.2.2. *IDatabase*

IDatabase is the main image-indexing program and image database creator. It can append the following image types: BMP (Windows Bitmap), TGA, TIFF, JPEG, JPEG 2000 [20], PNG, PCX, PGM, GIF, WMF, PCT and EPS. During the appending operation the original image is either direct-stream-copied to the indexing location within the database or the image is trans-coded into one of the following formats:

- JPEG (with a quality parameter QP)
- JPEG 2000 (with a compression factor = $1/QP$)
- TIFF
- PNG
- Windows BMP

Such a trans-coding capability enables us to form image databases with different QPs using JPEG or JPEG 2000. Hence the effects of LBR compression on retrieval-by-query performances can be examined.

IDatabase creates image or hybrid databases and it can append images to any of the database types. The rule for creating a hybrid database is the following: even if only one image is appended to an existing video database, it is then automatically changed to a hybrid database since it now contains both of the media types (video and image).

2.2.3. *DbEditor*

Once the media databases are initially created by the database creator applications, *DbEditor* application is designed to handle indexing and all the off-line issues for the existing databases.

Feature editing is the primary task of *DbEditor* application. This is basically needed for proper indexing of the multimedia databases. Hence *DbEditor* can add and remove features to/from any type of MUVIS database. Main functionalities of *DbEditor* can be listed as follows:

- Dynamic integration and management of feature extraction modules,
- Extracting new features or removing existing features of a database,
- Appending external video clips and images to any existing database,
- Removing existing video clips or images from the database,

- Copying items from a database to another while preserving the indexing structure of the target database,
- Previewing any video clip or image in a database,
- Statistical information of database and/or items in a database.

2.2.4. *MBrowser*

MBrowser is the main media retrieval and browser terminal, which supports any kind of database, image and video (including) types mentioned in the previous sections. In the most basic form, it has capabilities of an advanced multimedia player (or viewer) and a database browser. Furthermore, it allows users to reach any kind of multimedia easily, efficiently and in any of the hierarchic level that is designed for the video clips. *MBrowser* supports 4 levels of video display hierarchy: single frame (1st frame), shot frames (key-frames), scene frames and the entire video clip.

MBrowser has a built-in search and query engine, which is capable of finding multimedia primitives in a database and for any multimedia type that is similar to a queried media item (a video clip, an audio clip, a frame or an image). For video query, the video clip queried should first be appended to a MUVIS database that the query will be performed. There is no such necessity for images, any image (inclusive or exclusive) can be queried in a database. Retrieval is based on comparing the query feature vector(s) with the feature vectors of multimedia primitives available in the database. The comparison is based on the similarity measurement function implemented in the corresponding feature extraction module, which is integrated into MUVIS. Furthermore, *MBrowser* provides the following additional functionalities:

- Video summarization via scene detection,
- Key-frame browsing during video playback,
- Post-processing to the video frames,
- Random access support during video playback,
- Displaying information related to the extracted features of the active database,
- Graphical interface for the visual feature vectors of the images and video frames.

3. INDEXING FRAMEWORK IN MUVIS

In order to provide both visual and aural indexing schemes MUVIS supports both visual and aural feature extraction frameworks in such a way that feature extraction modules can be developed independently and integrated into MUVIS system exclusively (during the run-time). This is the basis of the framework structure, which therefore allows third parties to develop their own feature extraction modules and integrate into MUVIS without knowing the details of the MUVIS applications. The following sections explain in detail the audio and visual feature extraction schemes in MUVIS.

3.1. Visual Feature Extraction Framework: *FeX*

Visual feature extraction is mainly applied on two visual media types in MUVIS framework: video clips and images. Features of video clips are extracted from the key-frames of the video clips. During real-time recording phase, *AVDatabase* may optionally store the uncompressed key-frames of a video clip along with the video bit-stream. If not, *DbEditor* can extract the key-frames from the video bit-stream and store them in raw (YUV 4:2:0) format. Those key-frames are the INTRA frames in MPEG-4 or H.263 bit-stream. In most cases, a shot detection algorithm is used to select the INTRA frames during encoding but sometimes a forced-intra scheme might further be applied to prevent possible degradations. Image features on the other hand are extracted directly from 24-bit RGB frame buffer obtained by decoding the image.

MUVIS manages feature extraction in terms of explicit modules. Feature extraction algorithms can be implemented independently as modules, and then integrated into MUVIS framework via a specific interface called Feature Extraction Interface (*FeX* API). *DbEditor* is responsible for the management of the features in any kind of multimedia database. Additionally, *DbEditor* supports multiple sub-feature extraction with different parameters. All visual feature extraction algorithms should be implemented as a *Dynamic Link Library* (DLL) with respect to *FeX* API, and stored in an appropriate folder. *FeX* API provides the necessary handshaking and information flow between a MUVIS application and the feature extraction module.

The following feature extraction algorithms have been integrated so far into MUVIS framework using *FeX* API:

- HSV, RGB and YUV Color Histogram features.
- Gray Level Co-occurrence Matrix method for texture feature [12].
- Gabor Wavelet Transform method for texture feature [12].

More detailed information about *FeX* framework can be found in [14].

3.2. Audio Indexing and Retrieval Scheme

Recently a generic audio indexing and retrieval scheme have been successfully implemented in MUVIS. As shown in Figure 2 audio indexing is accomplished in several stages: Classification and Segmentation, Audio Framing within segments with certain types, Audio Feature Extraction (*AFeX*), Key-Framing via Minimum Spanning Tree (MST) Clustering and finally the indexing over the extracted Key-Frames (KFs).

3.2.1. Audio Classification and Segmentation

In order to achieve a better content analysis the first step is to perform classification and segmentation over the entire audio clip. In MUVIS system this operation is performed over AAC and MP3 bitstreams directly from the bitstream information. The method is generic and robust to several

capture and encoding parameters. More detailed information can be found in [17].

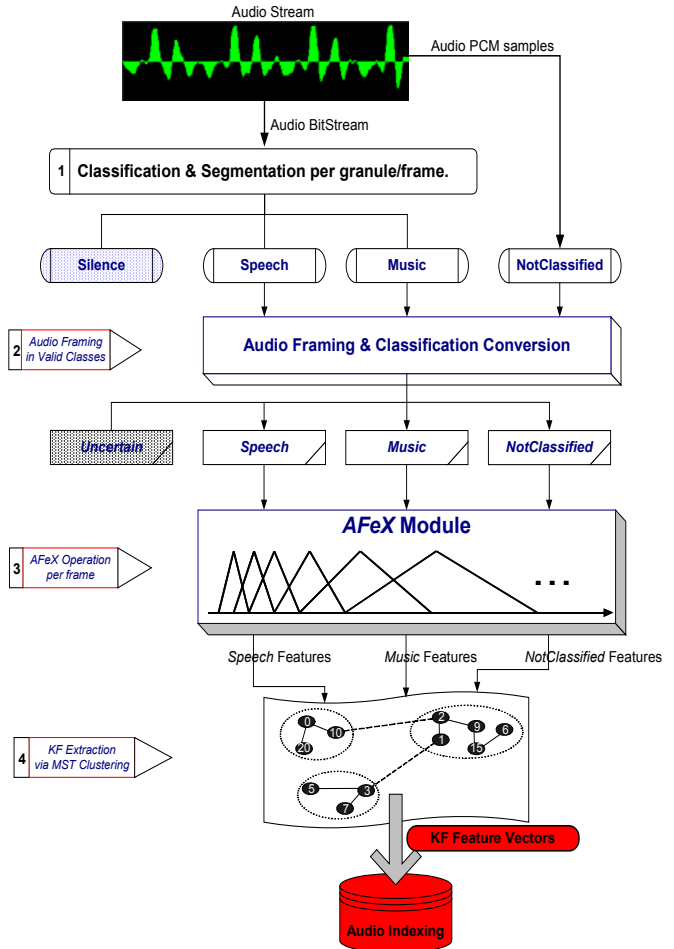


Figure 2: MUVIS Audio Indexing Operation Flowchart.

3.2.2. Audio Framing within Valid Audio Segments

There are three valid audio segments: *speech*, *music* and *NotClassified*. 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* and will be excluded from any *AFeX* operation and hence audio indexing process. Figure 3 shows an audio framing example. If the audio is in PCM or ADPCM format, all the audio frames in the entire clip will be assigned as *NotClassified* since the classification and segmentation algorithm for MP3 and AAC bit-streams is not applicable for such audio formats.

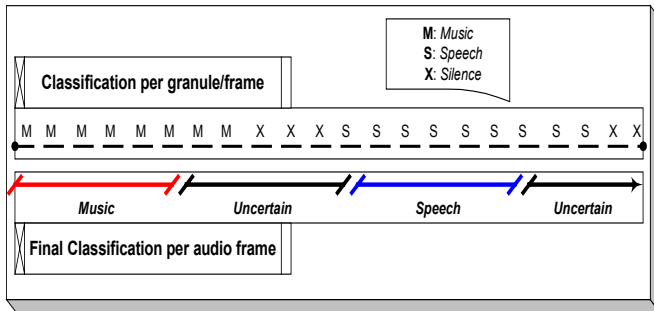


Figure 3: A sample audio classification conversion.

The removal of audio frames in *Uncertain* and *Silence* types from *AFeX* operation has two advantages: first there is no need for *AFeX* operation on silent frames since no content information can be retrieved from them. Similarly the *Uncertain* frames are mixed and hence most of the times they are transition frames (i.e. music to speech, speech to silence, etc.). Therefore, the feature extraction will result an unclear feature vector, which does not contain a clean content characteristics at all. The second advantage is the significant reduction of the number of audio frames from the indexing scheme. This will eventually reduce the computational time for following indexing operations such as audio feature extraction and Key-Framing. Most important of all this will further reduce the retrieval time of any aural query within a database.

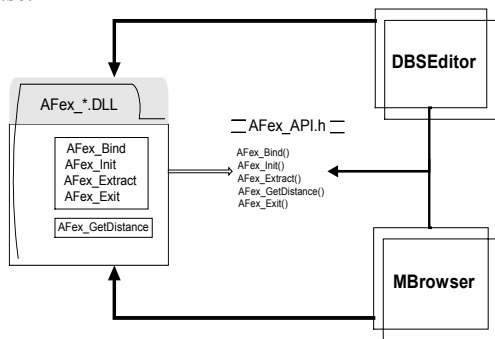


Figure 4: Basic AFeX Module interaction with MUVIS applications.

3.2.3. Audio Feature Extraction (*AFeX*) Framework

Once audio framing is completed, feature extraction is applied onto certain frames with a valid class types (*music*, *speech* or *NotClassified*) for indexing. *FeX* framework for visual feature extraction has been presented in section 3.1. A similar framework (so called *AFeX*) has been developed to accomplish audio feature extraction operations. Therefore, *AFeX* framework mainly supports dynamic audio feature extraction module integration for audio clips and such a framework shall form a common basis to compare, merge and experiment among several exclusive *AFeX* modules to develop new algorithms and to improve efficiency. Figure 4 shows the API functions and linkage between MUVIS

applications and a sample *AFeX* module. All audio feature extraction algorithms should be implemented as a *DLL* with respect to *AFeX* API, and stored in an appropriate folder. *AFeX* API provides the necessary handshaking and information flow between a MUVIS application and an *AFeX* module. Currently MFCC *AFeX* module is successfully implemented in MUVIS. MFCC stands for Mel-Frequency Cepstrum Coefficients [19] 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.

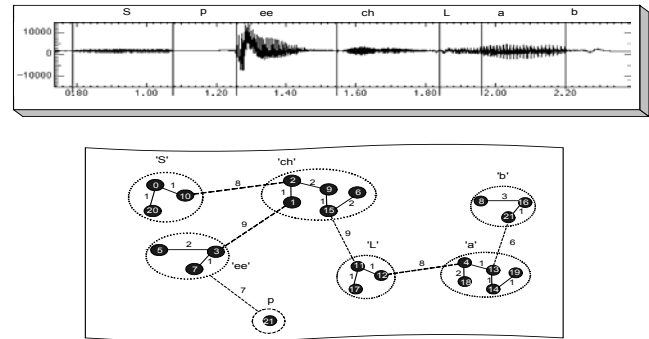


Figure 5: An illustrative clustering scheme for audio KF extraction.

3.2.4. Key-Framing via MST Clustering

The total 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 scheme (*FeX*) where only the visual feature vectors of the Key-Frames (KFs) are stored for indexing.

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 5. Here the problem is to determine the number of clusters that should be extracted over an entire clip. For this we define *KF rate* that is the ratio between the number of KFs over total number of valid frames. Once a practical *KF rate* is set, the number of clusters is 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 a Key-Framing

model that will reduce it for longer duration clips and increase it for shorter duration clips.

Once the number of KFs ($KFno$) is set, the audio frames are then clustered by using *Minimum Spanning Tree* (MST) clustering technique [18]. 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. There is however one practical problem during MST formation: the number of comparisons and distance calculations is proportional with the square of the number of nodes (valid frames) present. So this brings a practical maximum number (N_c^{MST}) of nodes for MST formation. Therefore, if the number of nodes is exceeding this practical node limit, then the audio frames are first separated into chunks, which contain N_c^{MST} nodes within the chunk c . For each chunk the MST clustering and associated Key-Framing process are performed and the feature vectors of the KFs are then stored for audio indexing.

3.2.5. Aural Retrieval-by-Query Scheme

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. As shown in Figure 6, 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 obtaining the similarity measurement. There is one particular exception of this: As shown in Figure 2, some of the segments of an audio clip might have *NotClassified* type due to some ambiguity in the classification scheme or simply the lack of the presence of a reliable classifier. Naturally this should not prevent the retrieval since such parts may include important content information. Therefore, the audio frames with *NotClassified* type are entirely included in the retrieval scheme and since their class types are unknown, they are to be compared with the all the frames, which have a valid class type. For instance, the similarity measurement for the audio frames of the queried clip in *speech* type will only be performed with the audio frames of the database clips having *speech* type and *NotClassified* type. This scheme based on classification will ensure to prevent any false (mismatched) query retrievals and reduce the computational time significantly.

In order to accomplish an audio based query within MUVIS, an audio clip is chosen in a multimedia database and queried through the database if the database includes at least one audio feature. 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 $0 \leq s < NoS$. 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 for 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 total sub-feature similarity distance ($D(s, f)$) between two clips. Figure 6 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:

i) 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 $N_Q^\circ(s, f)$ 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 $N_Q^\circ(s, f)$ 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, $N_Q(s, f) = N_Q^\circ(s, f) + N_Q^\circ(s, f)$ and the class mismatch penalization can be formulated as follows:

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

Equation 1

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

ii) 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 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 to 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 can be used for experimentation in order to find an optimum merging scheme for the audio features (and associated sub-features) available in the database. The following equation formalizes the calculation of QD_c .

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

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

$$QD_c = \sum_s \sum_f^{N_Q^f(s)} W(s, f) \cdot D(s, f)$$

Equation 2

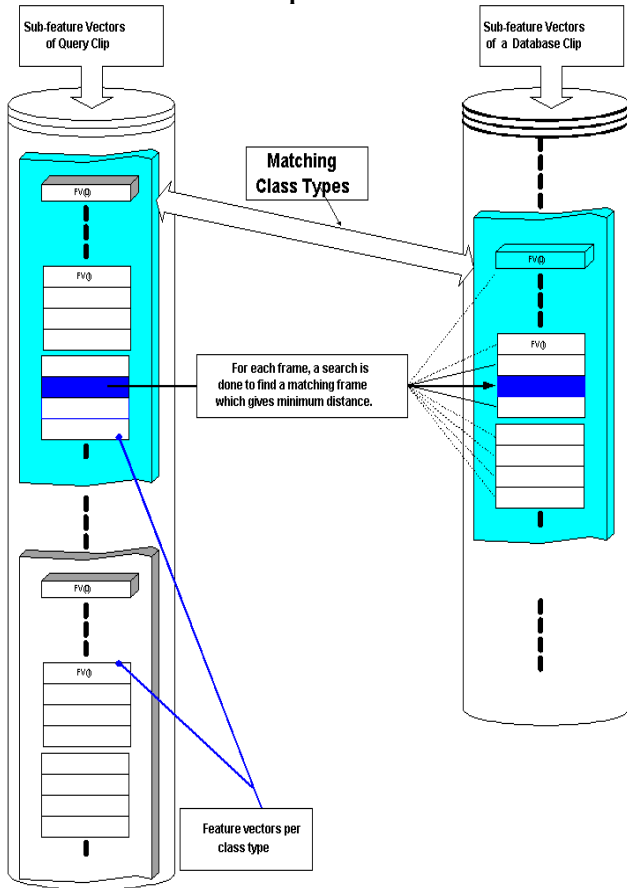


Figure 6: A class-based audio query illustration showing the distance calculation per audio frame.

Calculation of QD_c via Equation 2 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 are nothing in common with respect to a high-level content analysis such as their mismatching audio class types per segment.



Figure 7: MUVIS MBrowser GUI with playback of a video clip on the left and 12-best aural query results on the right side.

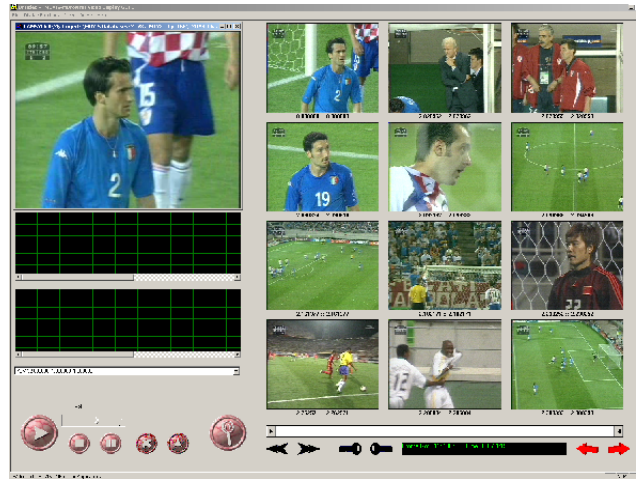


Figure 8: MUVIS MBrowser GUI with playback of a video clip on the left and 12-best aural query results on the right side.

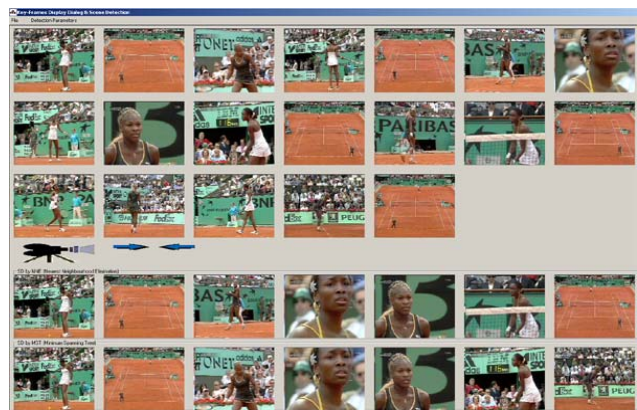


Figure 9: Video Summarization in MUVIS. Key-Frames are shown in the upper side and in the bottom side 7 Scene Frames, which are extracted by two different methods are shown.

4. REMARKS AND CONCLUSIONS

Figure 7 and Figure 8 show an aural query (query with audio features, i.e. MFCC) of a video clip from a hybrid database containing 185 video clips carrying MP3 audio. For both queries the first best 12 query results are shown. Figure 9 shows a video summarization scheme in MUVIS. The video is encoded in MPEG-4 in a way that the shot-frames are I-frames and used as KFs. Therefore, KFs are directly extracted from the bit-stream and then Scene Frames are further extracted using two different methods [7] and shown in the bottom side. This GUI provides random access through the KFs and hence the user can play the video between two KFs, so called *Region of Interest* (RoI) and then a partial query can be done within RoI through a multimedia database.

The current MUVIS system has been developed to bring a unified and generic solution for content-based multimedia indexing and retrieval problem. A generic solution is basically achieved by designing the whole system of applications, which are handling several multimedia items within several formats and types during their life-time starting from capturing till retrieved and rendered in such a way that the indexing mechanism is designed to be independent from the underlying media properties such as capturing and encoding parameters, file formats, encoder types, duration, etc. Therefore, such a generic indexing mechanism yields a media retrieval depending only on the content information regardless from such media properties. The framework covers a wide range of media types and formats for handling, indexing and retrieving. It supports browsing, hierarchic video representation and summarization. Most important of all, MUVIS framework supports integration of the aural and visual feature extraction algorithms explicitly. This brings a significant advantage for third parties to develop and test several feature extraction modules that are independent from the MUVIS applications.

Several retrieval experimental results show the effectiveness of the overall indexing and retrieval framework and the available *FeX* and *AFeX* modules. The indexing framework achieves a significant query performance and it provides a robust and generic solution for several multimedia types, capture parameters, coding methods, file formats and several other factors that MUVIS system supports. Furthermore, the MUVIS framework can be improved according to the feedbacks from the owners of the *FeX* and *AFeX* modules in order to provide more efficient and global platform as a generic test-bed for content-based multimedia indexing and retrieval area.

5. REFERENCES

[1] M. Gabbouj, S. Kiranyaz, K. Caglar, B. Cramariuc, F. Alaya Cheikh, O. Guldogan, and E. Karaoglu, "MUVIS: A Multimedia Browsing, Indexing and Retrieval System", *Proceedings of the IWDC 2002 Conference on Advanced Methods for Multimedia Signal Processing*, Capri, Italy, Sep. 2002.

- [2] 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.
- [3] J.R. Smith and Chang, "VisualSEEK: a fully automated content-based image query system", *ACM Multimedia*, Boston, Nov. 1996.
- [4] Virage. [URL:www.virage.com](http://www.virage.com)
- [5] 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.
- [6] ISO/IEC JTC1/SC29/WG11, "Overview of the MPEG-7 Standard Version 5.0", March 2001.
- [7] S. Kiranyaz, K. Caglar, B. Cramariuc, and M. Gabbouj, "Unsupervised Scene Change Detection Techniques In Feature Domain Via Clustering and Elimination", *Proceedings of the IWDC 2002 Conference on Advanced Methods for Multimedia Signal Processing*, Capri, Italy, September 2002.
- [8] ISO/IEC JTC1/SC29/WG11, "Coding of Moving Pictures and Audio: Overview of MPEG-4 Standard", V. 21, March 2002.
- [9] ITU-T Recommendation H.263, "Video Coding For Low Bit Rate Communication", February 1998.
- [10] ISO/IEC 13818-3:1997, Information Technology – Generic Coding of Moving Pictures and Associated Audio Information – Part3: Audio, 1997.
- [11] ISO/IEC CD 14496-3 Subpart4: 1998, Coding of Audiovisual Object Part3: Audio, 1998.
- [12] 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.
- [13] W. Y. Ma, B. S. Manjunath, "A Comparison of Wavelet Transform Features for Texture Image Annotation", *Proc. IEEE International Conf. On Image Processing*, 1995.
- [14] O. Guldogan, E. Guldogan, S. Kiranyaz, K. Caglar and M. Gabbouj, "Dynamic Integration of Explicit Feature Extraction Algorithms Into MUVIS Framework", *FINSIG 2003, Finnish Signal Processing Symposium*, Tampere, Finland 2003.
- [15] F. Alaya Cheikh, B. Cramariuc, C. Reynaud, M. Quinghong, B. Dragos-Adrian, B. Hnich, M. Gabbouj, P. Kerminen, T. Mäkinen and H. Jaakkola, "MUVIS: a system for content-based indexing and retrieval in large image databases", *Proceedings of the SPIE/EI'99 Conference on Storage and Retrieval for Image and Video Databases VII, Vol.3656*, San Jose, California, 26-29 January 1999.
- [16] F. Alaya Cheikh, B. Cramariuc, M. Partio, P. Reijonen, and M. Gabbouj, "Ordinal-Measure Based Shape Correspondence", *Eurasip Journal on Applied Signal Processing, Vol. 2002, No. 4*, April 2002.
- [17] S. Kiranyaz, M. Aubazac, M. Gabbouj, "Unsupervised Segmentation and Classification over MP3 and AAC Audio Bitstreams", *WIAMIS Workshop*, pp. 338-345, London, 2003.
- [18] Graham, R.L., and O. Hell, "On the history of the minimum spanning tree problem," *Ann. Hist. Comput.* 7, pp. 43-57. 1985.
- [19] L. R. Rabiner and B. H. Juang, "Fundamental of Speech Recognition", Prentice hall, 1993
- [20] ISO/IEC 15444-1:2000 Information technology -- JPEG 2000 image coding system.
- [21] ISO/IEC 14496-1:2001 N4270-1 "ISO Media File Format Specification" May, 2002.