

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

MUVIS is a series of CBIR systems. The first one 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, MUVIS has been reformed to become a PC-based framework, which supports indexing, browsing and querying of various multimedia types such as audio, video, audio/video interlaced and several image formats. MUVIS system allows real-time audio and video capturing, encoding by last generation codecs such as MPEG-4, H.263+, MP3 and AAC. It supports several audio/video file format such as AVI, MP4, MP3 and AAC. Furthermore, MUVIS system provides a well-defined interface for third parties to integrate their own feature extraction algorithms into the framework and for this reason it has recently been adopted by COST 211quat as COST framework for CBIR. In this paper, we describe the general system features with underlying applications and outline the main philosophy.

1. INTRODUCTION

During the recent decades, technological hardware and network improvements have caused a rapid increase in the size of digital visual information. Besides several benefits and usages, such massive digital visual information has brought about the problems of storage and management. In order to overcome such problems several content-based indexing and retrieval techniques and applications have been developed.

Existing indexing and retrieval systems such as MUVIS system [1], [15], [16], Photobook, VisualSeek, Virage, and VideoQ [2],[3],[4],[5] provide the framework and techniques for indexing and retrieving either image or video information. There are also several initiatives and frameworks 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, has 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,
- Video summarization via scene detection [7],
- Dynamic integration of explicit aural and visual feature extraction algorithms,
- Handling of several multimedia file formats such as Microsoft AVI, MP4 (MPEG-4 v1 and v2 file format), 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 SP, MP3, AAC, G721, G723. Uncompressed (raw) audio (PCM) and video (YUV, RGB) 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 3 we outline the system philosophy of MUVIS and the general MUVIS framework with underlying

applications. Section 3 presents the general indexing scheme in the MUVIS framework. In section 4, we demonstrate some experimental results on retrieval by query and we outline our conclusions and remarks.

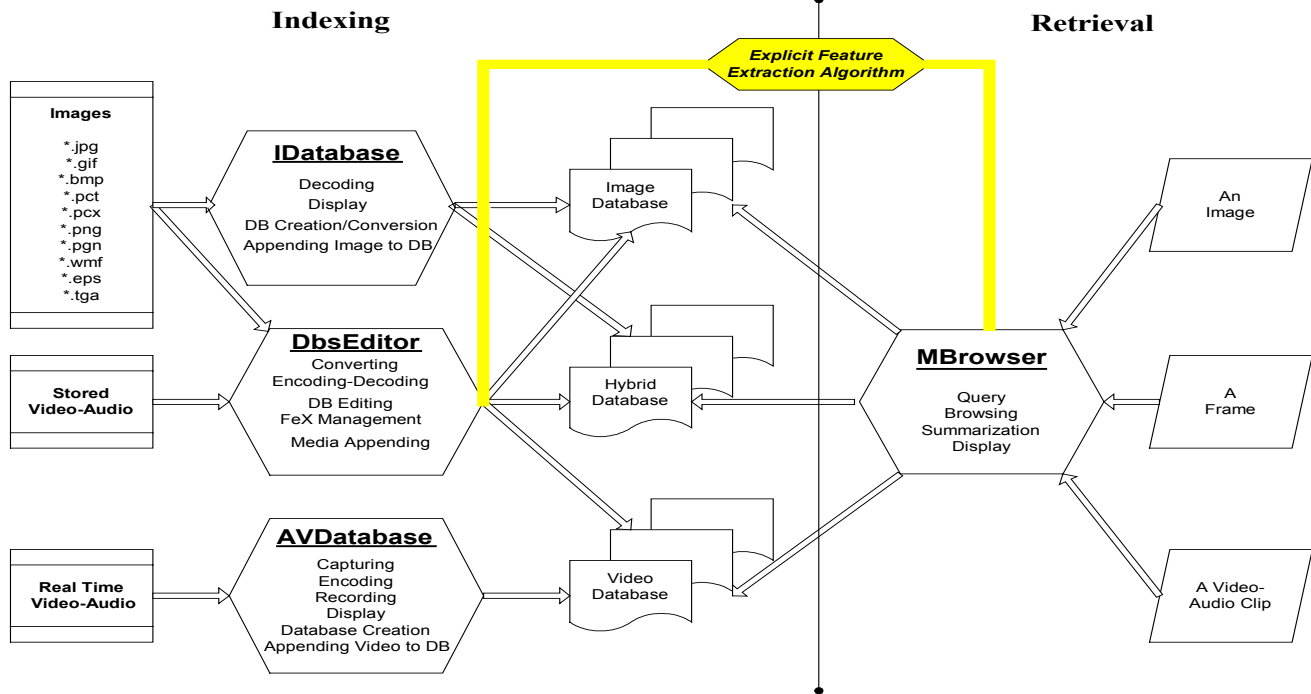


Figure 1: General structure of MUVIS framework

2. APPLICATIONS OF MUVIS FRAMEWORK

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. In the following subsections we review the basic features of each application while emphasizing their role in the overall system.

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 interlaced as in the way the underlying file format supports. 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) \rightarrow YUV 4:2:0
- RGB24 or RGB32
- YUY2 (YUYV) or UYVY

Then it is converted to YUV 4:2:0 format for encoding. Capturing parameters such as video frame-rate and frame size can be defined during recording phase. The video is then recorded using MPEG-4 simple profile video encoder [8], H.263+ video encoder [9] or in raw (YUV 4:2:0 or RGB24) format into any file format, which is suitable for video recording such as AVI and MP4. 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 time-stamp 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 and AAC. ADPCM encoders such as G721 and G723 can also be used for low complexity and furthermore, audio can also be recorded in raw (PCM 16b) format. Audio is then recorded into any file format, which is suitable for

audio recording such as MP3 and AAC or possibly interlaced with video, such as AVI and MP4. It is also feasible to store standalone audio information into AVI and MP4.

2.2. IDatabase

IDatabase is the main image-indexing program and image database creator. It can index the following image types: BMP (Windows Bitmap), TGA, TIFF, JPEG, PNG, PCX, PGM, GIF, WMF, PCT and EPS.

IDatabase creates image or hybrid databases and it can append images to any of MUVIS 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).

2.3. DbsEditor

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

Feature extraction is the primary task of *DbsEditor* application. This is basically needed for proper indexing of the multimedia databases. Hence *DbsEditor* can add and remove features to/from any type of MUVIS database. Main functionalities of *DbsEditor* 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,
- Preview of any video clip or image in a database,
- Statistical information of database and/or items in a database.

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-level 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, 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,
- Visualizations of feature vectors of the multimedia items.

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 Indexing and FeX Framework

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, *DbsEditor* 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). *DBsEditor* is responsible for the management of the features in any kind of multimedia database. Additionally, *DBsEditor* 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. Figure 2 summarizes the API functions and linkage between MUVIS applications and a sample 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].

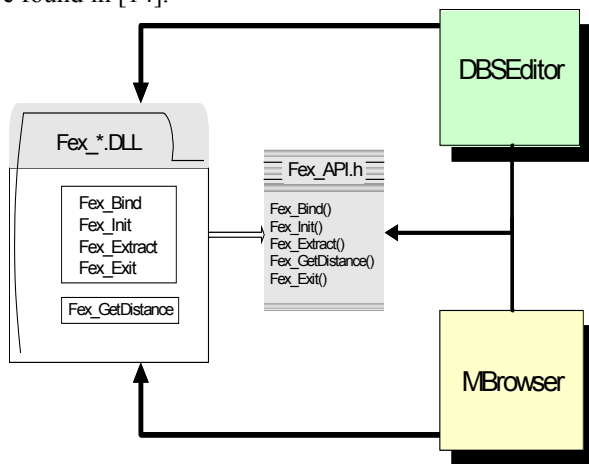


Figure 2: FeX module interaction with MUVIS applications

3.2. Audio Indexing and *AFeX* Framework

Recently audio indexing and retrieval scheme have been successfully implemented in MUVIS. As shown in Figure 3 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].

The output from the classification and segmentation is the segments with certain duration in time and a certain class type such as *speech*, *music* and *silence*. 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 and 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

Currently for ADPCM and PCM audio, this step is not performed and all the frames are assigned as *NotClassified*. Any *NotClassified* frame is due to feature extraction and similarity comparison with all the frames within the segments with any valid class types such as *speech*, *music* and *NotClassified*. There is an ongoing study to bring a classification and segmentation scheme for PCM and ADPCM audio in the near future.

3.2.2. Audio Framing within Valid Audio Segments

As mentioned in the previous section, 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 4 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* as mentioned before.

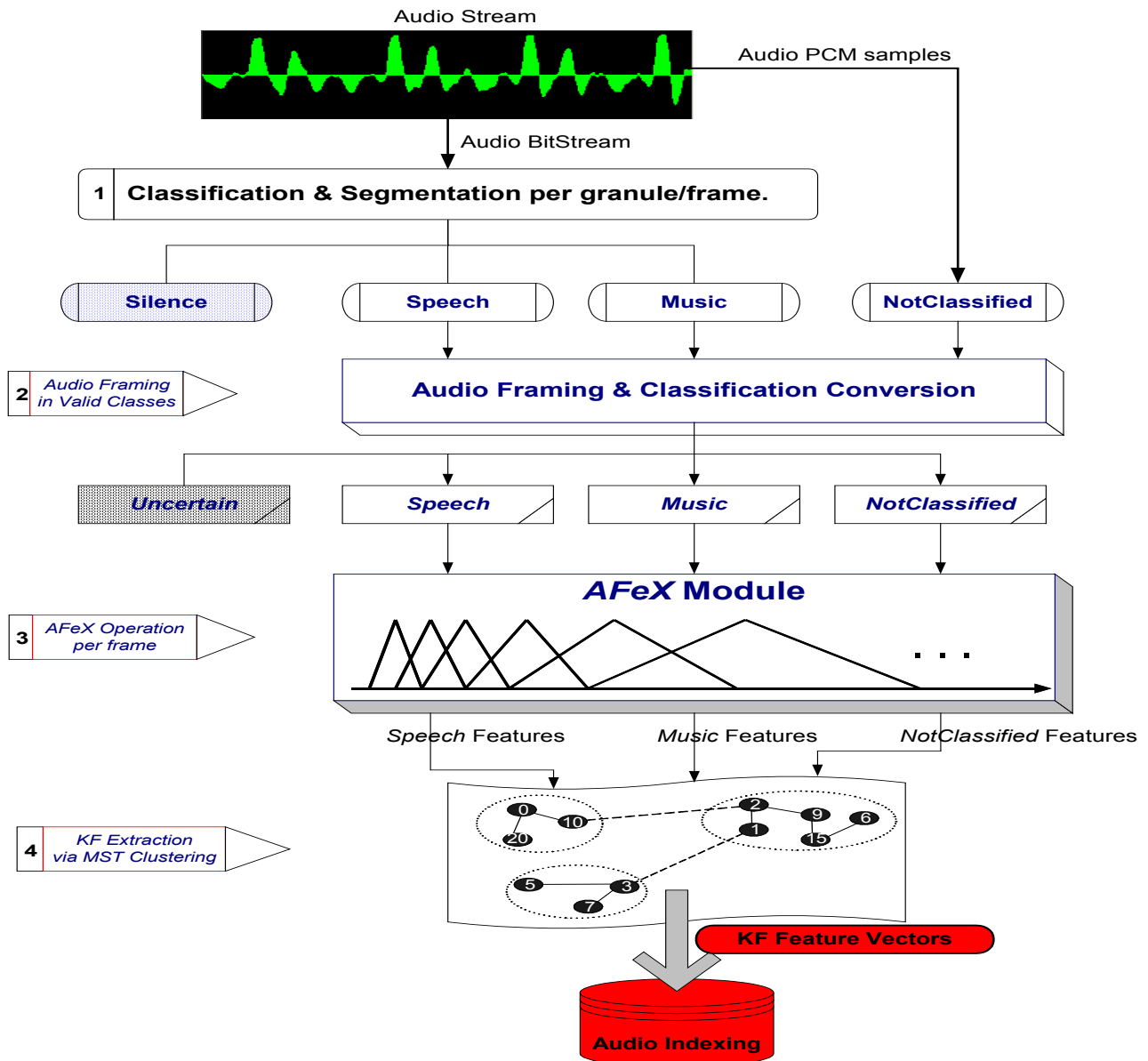


Figure 3: MUVIS Audio Indexing Operation Flowchart.

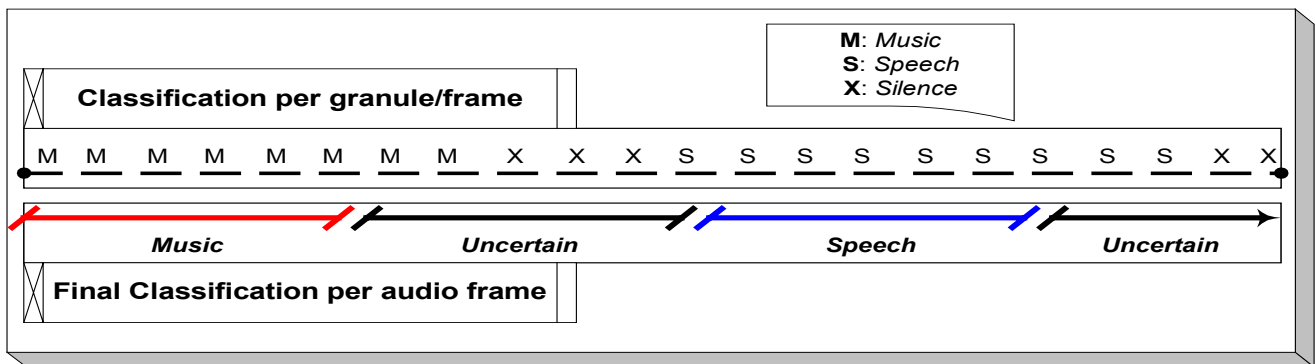


Figure 4: A sample audio classification conversion.

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 clip query within a database.

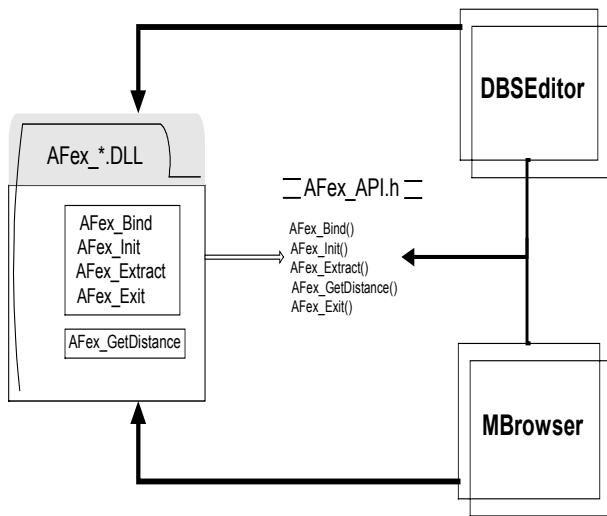


Figure 5: Basic AFeX Module interaction with MUVIS applications.

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 5 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. 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.).

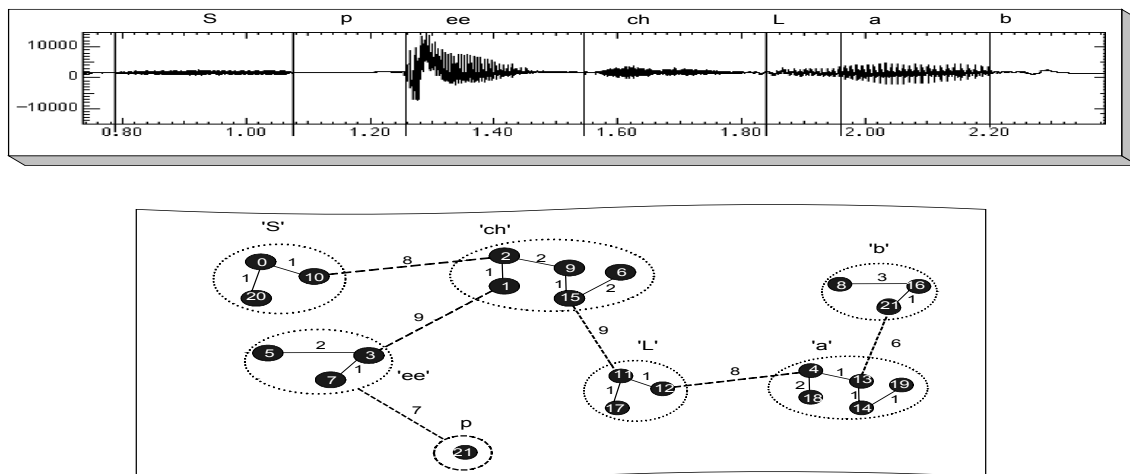


Figure 6: An illustrative clustering scheme for audio KF extraction.

3.2.4. 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 feature extraction but in aural case since there is no such a physical ‘frame’ structure and hence the audio is framed uniformly with some certain duration, we need to know features of each frame beforehand in order to make Key-Frame analysis. This is why *AFeX* operation is performed (over valid frames) first and then KFs are extracted.

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 ideal case example is shown in Figure 4. Here the problem is to determine the number of clusters that should be extracted over an entire 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 total number of valid frames. 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 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.

4. REMARKS AND CONCLUSIONS

As shown in Figure 7, an image database containing 1594 images is loaded and an image from this database is queried on the GUI window of *MBrowser*. The first best 12 query results are shown on the right side and the queried image is shown on the top-left side of the figure. The bottom-left corner shows a sketch of the feature vector of the queried image (up) and a sketch of the feature vector of the 3rd best (in top row, right-most) image. Figure 8 shows GUI window with a video playback on the top-left corner with its key-frames on the bottom-left corner. If requested, key-frames may be displayed simultaneously with the video playback. This video clip is also queried through a video database, which contains 302 video clips about sports, news, talk-shows, series, advertisements, etc. The first best 12 query results are shown on the right side.

MUVIS framework is designed to bring a unified and global solution to content-based multimedia indexing and retrieval problem. The unified solution is basically achieved by designing the whole system of applications, which are handling the media during its life-time starting from capturing till indexing in such a way that the media can be indexed as efficiently as possible. 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 *FeX* - *AFeX* indexing and retrieval framework and the available *FeX* - *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.

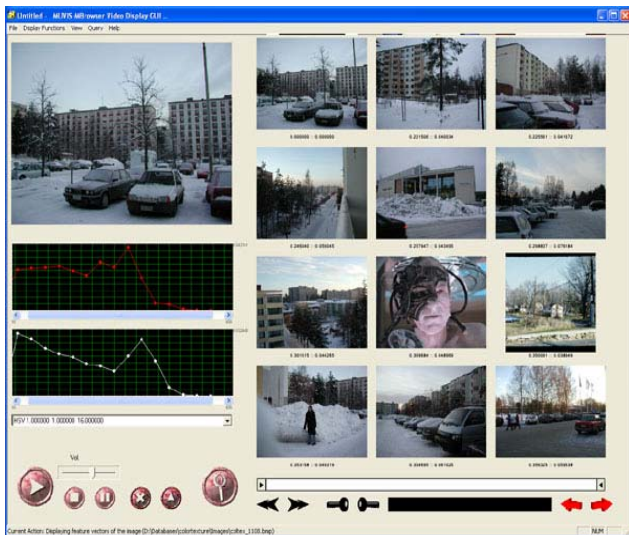


Figure 7: MBrowser GUI with an image database displayed and the query image on the top left corner.



Figure 8: MBrowser GUI with playback of a video clip (with its key-frames) query in a video database.

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