Survey of Compressed-Domain Features used in Audio-Visual Indexing and Analysis

Hualu Wang, Ajay Divakaran, Anthony Vetro, Shih-Fu Chang, and Huifang Sun

Wang and Chang are with Department of Electrical Engineering, Columbia University, New York, NY 10027.

Divakaran, Vetro, and Sun are with Advanced TV Laboratory, Mitsubishi Electric ITA, New Providence, NJ 07974.

Proposed running head: Survey on compressed-domain features in video…

Manuscript correspondence:

Hualu Wang
Email: hwang@ee.columbia.edu
Telephone: (212) 939-7156
FAX: (212) 932-9421

ABSTRACT

In this paper, we attempt to provide a comprehensive and high-level review of audio-visual features that can be extracted from the standard compressed domains, such as MPEG-1 and MPEG-2. The paper is motivated by the myriad of active research works in extraction and application of compressed-domain features in various fields, such as indexing, filtering, and manipulation. Compressed domain approaches avoid expensive computation and memory requirements involved in decoding and/or re-encoding. Selected features are categorized into four groups – spatial visual (e.g., color, texture, edge, shape), motion (e.g., motion field, trajectory), audio (e.g., energy, spectral features, pitch), and coding (e.g., bit rate, frame/block type). For each feature, we briefly discuss the extraction methods, computational complexity, potential effectiveness in applications, and possible limitations caused by compress-domain approaches. Finally, we briefly describe audio-visual features specified in the MPEG-7 standard and discuss the possibility of extracting them in the compressed domain.

Keywords: features, video indexing, audio indexing, compressed-domain, multimedia, compression, MPEG-1, MPEG-2, MPEG-7.

I. Introduction

Recent advances in computer, telecommunications, and consumer electronics industries have brought huge amount of multimedia information to a rapidly growing audience. More and more digital audio and video data are made available over the Internet. Traditional TV broadcast is moving into the digital and interactive era. People are starting to get high-speed network connections via DSL and cable modem. Multimedia content provides rich information to consumers, but also poses challenging problems of management, delivery, access, and retrieval because of its data size and complexity.

In recent years, there has been active research trying to address these problems and make multimedia information efficiently accessible to the user. Researchers from signal processing, computer vision, and other related fields have generated a large body of knowledge and techniques. These techniques generally fall into one of the following research areas.

- **Video indexing** – Research in this area aims at creating compact indices for large video databases and providing easy browsing and intelligent query mechanisms [3, 10, 33, 39]. Potential applications include multimedia databases, digital libraries, and web media portals.

- **Video filtering and abstraction** – Research in this area tries to generate an abstract version of the video content that is important or interesting by extracting key portions of

the video [46, 58]. This can be used for personalized video delivery, intelligent digital video recording devices, and video summaries for large multimedia archives.

- **Audio indexing and analysis** – Research in this relatively new area includes audio classification, audio indexing and retrieval, music retrieval, etc [41, 53, 71]. Efforts have also been made in combining audio information with visual information to help index and analyze video content.

Besides these general areas, there is also some research with more specific objectives, which exploits knowledge in various domains. This has produced some interesting applications such as event detection in sports programs, anchorperson detection in news, and so on. However, techniques proposed in these applications are often limited to their application domains.

In all these areas, one critical and challenging issue is to extract *features* from multimedia data. Low-level features are compact, mathematical representations of the physical properties of the video and audio data. They greatly reduce the amount of data to be analyzed, provide metrics for comparison, and serve as the foundation for indexing, analysis, high-level understanding, and classification. Much research work has been done to investigate various visual and audio features, their extraction methods, and their application in various domains. In fact, the ISO/IEC International Standard MPEG-7 (ISO/IEC JTC1/SC29/WG11 N3752) is currently developing a set of descriptors (i.e., features) and description schemes for the description of multimedia content.

In this paper we will give an overview of state-of-the-art algorithms and tools using video and audio features. In particular, we will investigate features that can be efficiently extracted from compressed video and audio data. The reasons that we are interested in compressed-domain features are as follows. First, much of the multimedia content available today is in compressed format already, and most of the new video and audio data produced and distributed will be in standardized, compressed format. Using compressed-domain features directly makes it possible to build efficient and real-time video indexing and analysis systems. Second, some features, such as motion information, are easier to extract from compressed data without the need of extra, expensive computation. Of course, most features can be obtained from uncompressed data as well, usually with a higher precision but at a much higher computational cost. In practical systems, tradeoff between efficiency and accuracy can be explored. Compressed-domain and uncompressed-domain approaches can also be combined. For example, the compressed-domain approach is used to select candidates while the uncompressed-domain approach is used to find the most accurate results.

In this paper we will review compressed-domain features in the following categories: spatial visual features (such as color and texture), motion features, coding features, and audio features. Fig. 1 illustrates the hierarchy of audio-visual features used in video/audio indexing and analysis. Examples are also given for commonly used features in different categories. Note that meta-information, although very important for video/audio indexing, is beyond the scope of this paper. In our review, we will focus on selected features that can be extracted or estimated directly in the compressed domain, without trying to be exhaustive. We will also discuss several representative applications using these features. The purpose is to show how these low-level

features can be extracted, integrated, and used effectively for basic video indexing tasks (e.g., shot boundary detection), as well as applications that need to detect structures, semantic-level concepts, or events from the video (e.g., sports video analysis). Again, we focus on compressed-domain approaches. Finally, we will summarize a subset of effective and relatively robust features, which can be estimated directly in the compressed domain, and compare qualitatively these features in terms of efficiency, effectiveness, and limitations. Fig. 2 provides a concise list of these features with brief summaries and indices pointing to respective sections of this paper. We will also give an overview of important visual and audio descriptors included in the current MPEG-7 committee draft and discuss the possibility of extracting these descriptors from compressed video and audio.

The paper is organized as follows. Section II gives a brief overview of the MPEG-1/2 video and audio standards. Sections III, IV, and V include reviews of spatial visual features, motion features, and video coding features, respectively. In Section VI we discuss some important applications using the video features in previous sections. Section VII investigates audio features and their applications. Section VIII summarizes and evaluates a set of representative video and audio features, and briefly discusses MPEG-7 visual and audio descriptors (Section VIII.5). Section IX concludes the paper and discusses future directions.

II. Overview of Multimedia Compression Standards

International standards such as MPEG-1 and MPEG-2 have become very successful in several industries, including telecommunications, broadcast, and entertainment. The amount of multimedia data in these formats is growing rapidly. Other standards like MPEG-4 and H.26x

share many fundamentals of video and audio coding with MPEG-1 and MPEG-2. Therefore, many features we discuss in this paper can be extended to those standards as well.

1. Video Compression Technology

Digital video needs to be compressed for the purpose of efficient storage and transmission. Video compression technology encompasses a wide range of research areas such as communications, information theory, image processing, computer vision, psychophysics, etc. Years of active research in these fields have culminated in a series of video coding standards like MPEG-1, MPEG-2, H.26x, and so on. These standards share some core techniques such as block-based transform coding, predictive coding, entropy encoding, motion-compensated interpolation, etc. The most important ones are block-based transform coding and motion compensation.

Block-based transform coding reduces the *spatial* redundancy in digital video (or digital images). The substantial correlation between neighboring pixels is greatly reduced in transform coefficients. These coefficients do not need to be coded with full accuracy and can be entropy-coded efficiently for compression. The 8x8-block discrete cosine transform (DCT) is most widely used for its near-optimal performance and high speed using fast algorithms. A typical encoding sequence using the DCT in shown in Fig. 3. Note that in video compression other techniques are also involved, so that the actual encoder diagram is much more complex. In MPEG-1 and MPEG-2 video, the DCT is also used to encode differential data and residue errors after motion compensation.

Block-based motion compensation significantly reduces the *temporal* redundancy in digital video, as illustrated in Fig. 4. A best match of the same dimension is found for each block in the current frame, so that only the difference (residue error) between the block and its match needs to be coded. In MPEG-1 and MPEG-2, backward and bi-directional motion compensations are also used. These techniques provide a much higher coding efficiency than encoding each frame without looking at its adjacent frames for similarities. The unit of motion compensation is usually 16x16 blocks, termed macroblocks (MB) in MPEG video. The frequency of motion compensation and how it is done are flexible to allow for the tradeoff between encoding complexity and performance.

2. MPEG-1 and MPEG-2 Video

MPEG-1 and MPEG-2 are defined in ISO/IEC International Standards 11172 and 13818, respectively. MPEG-1 and MPEG-2 video coding uses the DCT transform to reduce spatial redundancy and block-based motion compensation to reduce temporal redundancy. There are five layers in an MPEG video bit stream: group of pictures (GOP), picture, slice, macroblock (MB), and block.

MPEG-1 video [25] focuses on the coding of non-interlaced video at bit rates up to 1.5 Mbps, with a typical picture size of 352 x 240. MPEG-1 is optimized for CD-ROM applications, and has been widely used in Video CD (VCD) applications and general-purpose video storage and archiving.

MPEG-2 video [27] targets the coding of higher resolution video (e.g., 720 x 480 picture size) with fairly high quality at bit rates of 4 to 9 Mbps. It aims at providing CCIR/ITU-R quality

for NTSC, PAL, and SECAM, and supporting HDTV quality, at data rate above 10 Mbps, real-time transmission, and progressive and interlaced scan sources. MPEG-2 video has been successfully used in Digital Versatile Disc (DVD) and is the standard for future Digital TV broadcast. Although similar to MPEG-1 video in fundamental components, there are several new features of MPEG-2 video such as

- Frame-picture and field-picture structure (allowing interlaced video)
- More flexible and sophisticated motion compensation (frame/field/dual prime)
- 4:2:2 and 4:4:4 macroblock formats
- New scan for intra-block DCT coefficients
- Scalability

More details can be found in the standards [25, 27] and textbooks [21] on MPEG-1/2 video.

3. MPEG Audio

MPEG audio [26, 28, 29] aims at coding of generic audio signal, including speech and music. MPEG audio coders are *perceptual* subband coders that utilize the *auditory masking* phenomenon [44]. The encoder constantly analyzes the incoming audio signal and determines the so-called masking curve, the threshold under which additional noise will not be audible by the human auditory system. The input signal is split into a number of frequency subbands. Each subband signal is quantized in such a way that the quantization noise will not exceed the masking curve for that subband. Fig. 5 shows the basic structure of an MPEG-1 audio encoder.

MPEG-1 audio [26] provides single-channel and two-channel audio coding at 32, 44.1, and 48 kHz sampling rate. Three different layers (namely Layers I, II, and III) are defined. These layers represent a family of encoding algorithms with increasing encoder complexity, performance, and encoding/decoding delay. Layer III has the highest encoder complexity and the best coding performance, but the longest encoding/decoding delay. The predefined bit rates range from 32 to 448 kbps for Layer I, from 32 to 384 kbps for Layer II, and from 32 to 320 kbps for Layer III.

MPEG-2 audio BC [28] is a backward-compatible multi-channel extension to MPEG-1 audio. Bit rate ranges are extended to 1 Mbps. Audio sampling rates are also extended towards lower frequencies at 16, 22.05, and 24 kHz.

MPEG-2 audio AAC [29] is a non-backward-compatible, very high-quality audio coding standard for 1 to 48 channels at sampling rates of 8 to 96 kHz, with multi-channel, multi-lingual, and multi-program capabilities. Three profiles of AAC provide varying levels of coding complexity and scalability.

Due to its relatively low coding quality, applications of MPEG-1 Layer-I audio are usually limited to audio storage in solid-state circuitry, off-line storage and editing of audio, etc. MPEG-1 Layer-II audio has more applications in commercial products such as Video CDs and MPEG-1 encoders. So far the most successful MPEG audio standards is MPEG-1 Layer-III audio, also known as MP3. Recent explosion of web sites offering MP3 music clips, along with

the debut of portable MP3 players, has attracted much attention and shown a huge potential of this standard in commercial products and services.

MPEG-2 audio, especially AAC, is a relatively new comer to the audio entertainment world. The industry has already produced an array of widely deployed proprietary audio coding standards, which aim at providing multi-channel, high-quality surround sound for movie theaters and home entertainment systems. These standards include Dolby Digital (using Dolby AC-3 audio coding [15]), DTS, SDDS, etc, many of which are based on perceptual audio coding techniques.

III. Spatial Visual Features

In this section we review spatial visual features that can be extracted from MPEG compressed video. These features can be used to characterize individual video frames, as well as JPEG-compressed images. They can also be applied to a group of video frames to characterize their aggregate visual property. Motion features, which are temporal visual features, will be discussed in the next section.

1. DCT DC Image Sequence

DCT DC image sequence is an iconic version of the original video. Although at a lower resolution, it captures the key content and is very efficient for visual feature extraction. All DCT coefficients, including the DC value, are readily available for I-frames in MPEG video. For P- and B-frames, however, only the residue error after motion compensation is DCT transformed and coded. To get the DCT coefficients in motion-compensated frames, Chang and

Messershmidt [9] developed a compressed-domain algorithm that computes DCT coefficients in translated blocks. This is based on the fact that the DCT is a linear transform. The algorithm becomes very efficient if only part of the DCT coefficients (e.g., the DC coefficient) is needed. Yeo and Liu [63], as well as Shen and Delp [50], also proposed efficient algorithms to approximate DCT DC values for P- and B-frames.

Once the DCT DC image sequence is constructed, features can be extracted (e.g., vectors formed by the YUV values of the DC images, and used for shot boundary detection, key frame extraction, and other video indexing purposes [64, 10, 32]).

The DCT DC sequences sometimes give better or more robust results for tasks like shot boundary detection because of its low-pass filtering nature. Note that for MPEG-1 video and some MPEG-2 video, the 4:2:0 macroblock format is used, so that the color resolution is half the intensity resolution (i.e., 8 x 8 sub-sampled for intensity, 16 x 16 sub-sampled for chrominance).

2. Color

Color features, such as color histogram, have been proved to be effective in image and video indexing and retrieval. For compressed video, color features can be extracted from the DCT DC sequence described above or other forms of progressively decompressed sequences. Sometimes intensity values are used instead of colors (i.e., the chrominance components are ignored), but in general intensity-only features offer much less information and are not as effective as color features for video indexing.

In image and video indexing, colors are frequently converted from RGB values to other perceptual color spaces, such as HSV, YUV, YCbCr, etc., because they are closer to human's color perception models. In MPEG video, colors are converted to YCbCr components. Therefore, it is easy to extract chrominance information and use them to compute approximate hue and saturation values if needed.

Tan et al. [55] use the absolute difference of *block color histograms* as the dissimilarity measure between two video frames. Dynamic programming is used to compare video clips of different length. DC sequences are used in their experiments of video query by an example clip.

Won et al. [62] use DC values of Y, Cb, and Cr, along with other features, to form a feature vector for each video frame for shot boundary detection purposes. The feature vector consists of hue histogram, luminance histogram, and macroblock edge types. Cb and Cr values are used to approximate hue and saturation of colors for the computation of the hue histogram. The *hue histogram* consists of six bins, representing the hues of six pure colors: red, yellow, green, cyan, blue, and magenta. To compute the hue histogram, hue/saturation values of colors are compared with each of the pure colors, and values are added to the corresponding bin inverse to the hue/saturation distance from that pure color. Thus, the hue histogram indicates the dominant hue, if there is any, in a video frame.

Other researchers use YUV color histogram from DC sequences to detect video shot boundaries, either by *histogram intersection* [64] or direct pair-wise comparison [10]. *Intensity*

histogram, combined with *projection histograms* (computed by projecting the intensity of a video frame on x and y axis), has also been used for shot boundary detection [42].

3. Texture and Edge

Textures and edges are fine structures in images and video frames, and require pixel-level processing to extract them. Unfortunately, this is usually not possible in compressed video without significant amount of decoding. However, because textures and edges correspond to mid-to-high-frequency signals in the DCT frequency domain, it is possible to obtain some level of texture and edge information by analyzing the frequency components (i.e., DCT coefficients). Coefficients in the 8 x 8 DCT block can be classified into frequency bands that roughly correspond to smooth areas, horizontal and vertical edges, and noisy areas [23].

Bao et al. [5] use *energy histogram* of low-frequency DCT coefficients as the matching feature of video frames to detect shot transitions. This feature can be seen as a rough description of the global texture pattern in the video frame.

Zhong et al. [72] compute *sums of amplitude* of AC coefficients in the first row and first column of the DCT coefficients matrix and use them as the criteria for initial segmentation of caption blocks in compressed video. This works because rapid changes in intensity (in both directions) introduced by character lines raise the energy level in the corresponding DCT frequency bands. In other words, caption blocks have a distinctive texture pattern that is reflected in the DCT coefficients.

There also have been some efforts on edge extraction from compressed video. Shen and Sethi [49] show that *AC energy* of DCT blocks can be used to detect areas of high activity. Observing that DCT AC coefficients are linear combinations of pixel intensity values in the 8 x 8 block, they use the ideal step-edge model and derive approximate *edge orientation, offset, and strength* parameters using only DCT AC coefficients. The result is coarse edge segments that can be used for video indexing tasks like shot boundary detection based on the change ratio of edge maps [67]. They also investigate *convolution-based edge detection* in compressed images by merging symmetric-kernel convolution with the IDCT procedures [48]. Compared with conventional convolution-based edge detection, a speedup of 3 to 10 times is achieved with comparable results. However, large convolution kernels (e.g., 17 x 17 pixels) are necessary to reduce the artifacts introduced by JPEG and MPEG compression schemes. Therefore, this method becomes less efficient and can be applied to only a small subset of MPEG video frames due to the computational complexity.

Song and Ra [51] use *edge block energy* to classify DCT blocks into edge blocks and non-edge blocks. Using the 3 x 3 Sobel operators, they show that the energy of the resulted gradient block can be approximated using the first few DCT AC coefficients. The result is a rough block edge map that is $1/64^{th}$ the size of the video frame, which indicate high-activity areas.

4. DCT Coefficients

DCT coefficients are sometimes used directly in applications such as shot boundary detection. DCT coefficients are readily available in I-frames, but not in P- and B-frames. Additional computation (full or partial decoding) is needed to obtain DCT coefficients in P- and

B-frames. Note that DCT coefficients of low-intensity blocks are prone to coding noise, which should be considered when using them for indexing or matching.

a. Correlation between DCT coefficients of two frames

Arman et al. [4] use a *subset of DCT coefficients* of a subset of blocks to form feature vectors for each frame. Then the *angles* between the feature vectors of different frames are used to measure their similarity. They use this method to detect shot boundaries and show its effectiveness for cut detection. The computational complexity is relatively high, because inner product is involved in the calculation.

b. DCT block difference

Zhang et al. [68] compare the *relative difference* of all coefficients in a DCT block to measure the similarity between two DCT blocks. A cut is detected if a large amount of blocks have changed significantly in terms of DCT block difference. This method involves less computation than the above one.

c. Variance of DCT DC coefficients

Meng et al. [37] use this feature to measure the variation of gray level intensity in I- and P-frames. Gradual transitions like dissolve can be detected by identifying a characteristic parabolic curve between two relatively flat variances.

IV. Motion Features

In compressed video, motion information is captured in some form like motion vectors in MPEG video. However, motion vectors are just a rough and sparse approximation to real optical flows, and are prone to be inaccurate when used to indicate real motion of macroblocks.

Extra care should be taken when using motion vectors. In MPEG video frames, the following areas are most likely to have erroneous motion vectors: a) boundary blocks, and, b) large smooth areas. Usually, some kind of morphological or median filter should be applied to the motion vector field to remove anomalies, before they are used for analysis and indexing. Also, motion vectors are more sensitive to noise in magnitude than in direction. Therefore, a median filtering based on magnitude is usually sufficient. More sophisticated methods have also been proposed using *reliability metrics* to get rid of erroneous motion vectors [66].

1. Motion Information Extracted from Motion Vectors

Motion information can be extracted for block, regions, objects, and whole video frames for motion query and object/region tracking.

a. Global motion field

Ardizzone et al. [2] use *global motion field* to model the global motion in video sequences for similarity query. The idea is to divide a frame into 4 or 16 quadrants with each quadrant a motion feature in terms of magnitude and direction. For magnitude, average value is used, and the query can be specified by high or low magnitude. For directional feature, either average or dominant motion vector direction (such as the largest bin in angle histogram), as well as angle histogram (90 directions, 4 degree each) is computed. Note that the number of bins

should not be too large, because MPEG motion vectors are sparse in inter-coded frames. The authors have also derived similar motion features using optical flows in the pixel domain [3].

b. Motion-based segmentation

Besides visual features like color and texture, motion information can be used to segment inter-coded video frames. This is usually very efficient compared with spatial feature-based methods. In the case that a large object moves over a uniform background, a simple binarization based on the magnitude of motion vectors could extract a reasonable foreground region, approximated by macroblocks. Otherwise, some form of clustering technique is necessary. Also, as mentioned above, motion vectors are usually pre-processed to remove anomalies.

Ardizzone et al. [2] use a *sequential labeling* method to segment video frames based on their similarity of motion vectors (both magnitude and angle). They also use a *clustering* method to extract regions with dominant motions using histograms of motion vector magnitudes. Each dominant region is then defined by size and average motion.

Eng and Ma [16] use an adaptive median filter to improve MPEG motion vector accuracy and propose a *unbiased fuzzy clustering* technique to extract dominant video objects.

c. Block-level motion analysis

Dimitrova and Golshani [12] use the MPEG motion vectors in P- and B-frames to approximate the movement of macroblocks. *Macroblock tracing* is used to retrieve low-level motion information. In the middle-level motion analysis, averaging or clustering method is used

to extract object motion for rigid and non-rigid objects, respectively. At the high-level analysis, a set of trajectories is associated with an activity defined using domain knowledge.

Kobla et al. [34] propose *flow estimation* that adopts similar ideas. It uses motion vectors in MPEG P- and B-frames to estimate the motion (termed *flow*) in every frame in terms of backward-predicted motion vectors, regardless of the original prediction mode. For example, in MPEG video with only I- and P-frames, the flow of each frame (except the last P-frame in the GOP) can be estimated by reversing the forward-predicted motion vectors. If B-frames are present, one can estimate the flow between the B-frame and its reference frame (P-frame or I-frame) first, by reversing forward-predicted motion vectors, or using backward-predicted motion vectors directly. Flows between two B-frames can then be derived based on their respective flows from the same reference frame.

d. Representation and matching of motion trajectories

Dimitrova and Golshani [12] use several representations for trajectories for various matching purposes as follows: exact motion trajectory (start position and trajectory coordinates), B-spline curves, chain code (for approximate matching), and differential chain code (for qualitative matching). Wavelet transforms can also be used to generate a multi-resolution representation of trajectories.

e. Accumulated motion

Accumulated motion at the global or object level has been used in several applications as well. For example, such motion features have been used to detect important events in special applications like sports.

Saur et al. [46] use the *accumulated camera panning motion* to detect fast break in basketball games. The accumulation is reset to zero when the motion changes direction. A peak in the accumulated panning motion is a candidate for fast breaks.

2. Camera Operation and Object Motion

Camera motion is an important feature in video indexing. Foreground moving objects can be extracted by differentiating camera motion and object motion. Camera motion should be taken into account in shot segmentation. Otherwise, there will be false alarms of shot boundaries. Furthermore, in some specific applications, models of camera motion can be used to detect important events (e.g., fast breaks in basketball games).

Camera operation usually causes a global and dominant motion flow in the video sequences. However, when dominant motion of a large object is present concurrently, it changes the global motion field and camera operation estimation becomes less reliable. One solution is to remove the outliers of motion vectors that do not fit well with the global motion models, and iterate the estimation process of camera motion.

a. Common camera operations

Various basic camera operations used in video production have been defined. A typical set consists of *pan, tilt, zoom, dolly, track, and boom,* as described in [1]. A slightly different set is defined in [56] as pan, tilt, zoom, swing, and translation, in which pan, tilt, and swing are the rotation around the x-, y-, and z-axis, respectively. In many applications, however, only pan, tilt, and zoom factors are considered.

b. Camera operation estimation based on physical models

These methods start with a physical model of the camera in the 3-D space, with some kind of projection that maps the object onto the camera's image plane. They estimate the parameters involved in the projected model, and the camera operations can be derived from these parameters. Usually some iterative algorithm is applied.

Tan et al. [56, 57] propose a camera motion estimation method based on the physical model of the camera and 3-D coordinate transforms. The camera operation is modeled as a combination of *rotations* about the three axes and a *translation* of the coordinates. The *zoom* operation is modeled as a scaling function of object projection onto the image plane. The object is assumed to be a rigid body. Under the assumptions that (1) the rotations are small (or a high sampling rate), and (2) the translation is minimal (or a high sampling rate), the model can be reduced to a *6-paramter* one.

In [56], the parameters are initially estimated using 4 pairs of pixel correspondences, and then refined using a recursive algorithm (the Kalman filter). In [57], motion vectors of compressed video are used as correspondences. The mean and variance of the estimated prediction error is calculated. Motion vectors that fall beyond a certain range are declared as outliers, and are excluded in the next iteration of the parameter estimation process. Experiments show that usually 2 or 3 iterations are sufficient for convergence. A simplified and faster version of the above algorithm, which is a *3-parameter* one, considers only pan, tilt, and zoom factors [57]. In this case a closed form solution exists for the estimation of the three parameters. Note that this simplified model takes out the perspective distortion factors in the original model, so

that the scene is assumed to be planer, which is approximately true when the object is far away from the camera.

Tse and Baker [59] model global motion with two parameters, a *zoom* factor and a *2-D pan* factor. The camera is modeled as a projective plane and the motion of the camera is small between frames, hence the pan effect is a displacement of objects in successive frames. The method assumes that motion vectors are mainly caused by global motion. Removing motion vectors that are not consistent with the estimated motion helps refine the estimation. This method is similar to the simplified model in [57].

Taniguchi, et al. [58] estimate camera motion to reconstruct panoramic icons for video browsing purposes. They use simple *pan and tilt* to model camera operations and use the least square error method to estimate the camera motions. They propose several criteria to validate the estimated parameters as follows: duration (the camera motion lasts for more than a specific period of time), smoothness of the camera operation, and goodness of fit (MSE minimized by camera operation compensation should be significantly smaller).

Bergen et al. [6] propose a hierarchical motion estimation framework in which various motion models can be used. The *affine model* is a 6-parameter motion model that is suitable for situations where the camera is far away from the background. Meng and Chang [36] use it for camera motion estimation. The idea is to minimize the estimated motion and the compensated one in the least square error sense. Pan and zoom factors can be derived using the six estimated

parameters. Patti et al. [43] also use this affine model for dominant global motion estimation in the application of extracting high quality stills from interlaced video.

c. Camera operation estimation based on dominant motion

These methods estimate camera operations by looking at the motion vectors directly. They do not adopt explicit physical camera models like those mentioned above. Usually different operations (e.g., pan and zoom) are estimated separately.

Zhang et al. [68] use sum of difference between motion vectors and the *modal vector* to determine pan and tilt. Zooming is detected by the *change of signs* of motion vectors across the center of zoom, as zooming often results in a pattern of many motion vectors pointing to or away from the zooming center. The problem is that a combination of zoom and pan is difficult to detect, since it considers them separately. Large object motion may also cause significant noise. One solution is to separate regions of different motion. However, if the moving object covers most of the frame, this will still be a problem. To quantitatively measure the camera operations, the method proposed by Tse and Baker [59] is used for zoom and pan.

Akutsu et al. [1] investigate video indexing by motion vectors. They first used a block matching method to estimate motion vectors similar to most MPEG encoders. Then they consider the camera operations separately and model each type of operation with different features of optical flows such as *vector convergence point*, and *vector magnitude*. Hough transform is used to estimate the convergence points, and a 3-parameter model is used to extract the magnitude of camera motions.

Kobla et al. [33] estimate camera pan and tilt by detecting dominant motion in the video frames. Motion vectors are counted to generate *directional histogram* (in 8 directions), and a dominant direction of motion is declared if the largest bin is at least twice as big as the second largest one. For zoom detection, the existence of *Focus of Contraction* (FOC) or *Focus of Expansion* (FOE) is detected. It is a simple voting method by extending each motion vector and voting for all the macroblocks it passes through. In addition, it also assumes that the motion vectors near the FOE and FOC are small and the magnitude of motion vectors increases with the distance to the FOE/FOC.

3. Statistical Information of Motion Vectors

Rather than estimating camera motions from the motion vectors, the following work measured the statistical features of the motion vectors and used them for indexing.

a. Motion smoothness

Akutsu et al. [1] define *motion smoothness* as the ratio of blocks that have significant motion vectors over blocks whose motion vector has changed significantly. They used this parameter along with inter-frame changes to detect shot boundaries.

b. Amount of motion activity in video frames

Divakaran and Sun [14] uses *average motion vector magnitude* as one of the features of a video frame. They then use this average value to threshold all the motion vectors in the frame and count the numbers of short, medium, and long *runs of zeros*, in raster-scan order. These numbers indirectly expresses the size, shape, and number of moving objects and are good indicators of the motion feature of the video frame. For example, a frame with a single large object can be easily distinguished from a frame with several small objects.

Wolf [61] uses the *amount of motion* in video frames to help select key frames in a video sequence. Although in [61] motion is estimated using optical flow, the principle can be applied to MPEG video using motion vectors. The amount of motion is defined as the summation of magnitude of motion vector components in a frame, and a curve is drawn with respect to frame numbers. Key frames are then selected by detecting valleys between large peaks in this curve, which correspond to relatively still frames between two video segments with large amount of motion.

c. Motion histogram

Motion histogram is a compact representation of global or regional motion in video frames. Either direction or magnitude can be used as the histogram index. Kobla et al [33] and Ardizzone et al [2] have used *motion vector histograms* extensively for the detection of camera pan, tilt, and many other purposes.

Although working in the uncompressed domain, Davis [11] also uses motion histograms to help recognize simple human movements such as left arm fanning up, two arms fanning up, and crouching down. The *directional histogram* for each body region has twelve bins (30 degree each), and the feature vector is a concatenation of the histograms of different body regions. The motion vectors are derived from so called Motion History Image (MHI), which is basically the accumulation of differencing images of the silhouettes of human body.

V. Video Coding Features

Video coding features are simple, but readily available information that can be easily extracted while parsing the video bit stream. In some cases, such simple features can be quite useful. Since MPEG only standardizes the decoding part, coding features may be sensitive to the actual encoder used.

1. Macroblock Type Information

Macroblocks in I-frames are all intra-coded. P-frames have forward-predicted macroblocks, as well as intra-coded and skipped ones. B-frames have the most complex situation, in which each macroblock is one of the five possible types: forward-predicted, backward-predicted, bi-directional predicted, intra-coded, and skipped. The numbers of these types of macroblocks, as well as the ratios between them, in P- and B-frames provide useful information for video indexing and analysis purposes.

Kobla et al. [30] use the *numbers* of forward-predicted and backward-predicted macroblocks in B-frames to determine if a slow-motion replay is present in MPEG video. In an earlier paper [33], macroblock type information is also used to detect shot boundaries.

Nang et al. [38] compare the macroblock types of the same macroblock position in adjacent B-frames. *Macroblock type changes* are listed exhaustively and assigned different dissimilarity values. Accumulation of all the macroblock dissimilarity values in one frame is used to detect shot boundaries.

Zhang et al. [68] use the *ratio* of the number of intra-coded macroblocks over the number of valid motion vectors in a P- or B-frame to determine abrupt shot changes that occur in inter-coded frames. Similarly, Meng et al. [37] use the *ratio* between the number of intra-coded macroblocks and the number of forward-predicted macroblocks to detect scene changes in P and B frames.

Saur et al. [46] compare the *number* of intra-coded blocks in P-frames with the *magnitude of motion* to determine if the frame is wide-angle or close-up in typical basketball video. A shot is classified by classifying all P-frames in it and a majority voting. The intuition is that in a close-up shot there are often large objects (e.g., players) entering or leaving the view, causing a significant number of macroblocks in P-frames to be intra-coded. This number is large compared with the magnitude of motion in a close-up shot.

2. Bit Rate Information

It has been observed that abrupt changes in video streams cause bit-rate variation in the encoded MPEG video. Feng et al. [18] form a *bit-image* for each frame using the number of bits taken to encode each macroblock. Shot boundary is detected by comparing bit-images using a fixed threshold. Divakaran et al. [13] and Boccignone et al. [7] combine this approach with DC image sequence based method for more robust shot boundary detection.

Frame bit rate, combined with macroblock and motion information, is also used by Kobla et al. [30] to detect slow-motion replay in sports videos.

VI. Applications Using Visual and Motion Features

In this section we will discuss some common applications that use the visual and motion features summarized in the previous sections. The purpose is to show how these features can be integrated and applied to specific compressed-domain video indexing and analysis problems.

1. Shot Boundary Detection

A shot is the basic unit in video production. A video sequence is a series of edited video shots. The transition between shots usually corresponds to a change of subject, scene, camera angle, or view. Therefore, it is very natural to use shots as the unit for video indexing and analysis, and the first step in these applications is to segment the long video sequence into video shots. This specific task of detecting transition of video shots is usually termed shot boundary detection, scene change detection, video segmentation, etc. Throughout this paper we use the term *shot boundary detection*. Note that we will focus more on gradual transition detection in the next section.

There has been a lot of work on shot boundary detection, especially in the uncompressed domain. Most work has been focusing on *pixel difference*, *intensity statistics comparison*, *histogram distance*, *edge difference*, and *motion information*. A review paper on these techniques with experimental results in terms of precision-recall graphs can be found in [8]. Among these methods, histogram-based ones have been consistently reliable, while DCT coefficient-based ones give the lowest precision. Motion information based methods are somewhere in between. In a recent review paper, Lienhart [35] compares four major shot boundary detection algorithms, which include fade and dissolve detection. Extensive experimental results also favor the *color*

histogram based method [8] for shot boundary detection, instead of the computationally expensive *edge-change-ratio* method [67].

Researchers have also investigated compressed-domain shot boundary detection techniques. One obvious approach is to apply uncompressed-domain techniques to DCT DC image sequences using approximated features. Techniques using pixel difference, intensity statistics, and histograms are still effective on DC images with some level of performance degradation. Experimental results of shot boundary detection techniques on DC images and uncompressed-images are compared in [20], using statistical performance metrics.

There are also shot boundary detection algorithms specifically proposed for MPEG compressed video. Zhang et al. [68] apply a *multiple-pass* and *multiple-comparison* technique to video segmentation. Multiple passes use I-frame *DCT information* to locate the rough location of scene changes, and the second pass use P- and B-frame *motion vector* information to pinpoint the location. Multiple comparison uses double threshold to detect abrupt and gradual shot transitions.

Chen et al. [10] use a feature vector extracted from DC image sequences called *generalized trace (GT)* for shot boundary detection purposes. The vector consists of YUV color histogram intersections (between consecutive frames), standard deviation of YUV, numbers of three types of macroblocks (intra-coded, forward-predicted, backward-predicted), and flags for frame types. Differential GT has been used for abrupt shot transition detection. More robust result is achieved using classification methods.

Meng et al. [37] use *macroblock type counts* in P- and B-frames to detect shot boundaries. Kobla et al [33] use similar *macroblock type information* for shot boundary detection in a more elaborate fashion. A DCT validation step is added to account for situations where there is little motion around shot boundaries, in which macroblock type information itself is insufficient. Two adjacent I-frames are compared using their DC values, and blocks that change significantly are counted and then compared with a threshold.

Other proposed techniques examine *macroblock type changes* [38], *motion vector changes* [19], and *macroblock bit allocation* information [13] to detect shot boundaries.

2. Gradual Transition Detection

Videos in commercial TV programs are rich in editing effects, such as fade in, fade out, dissolve, wipe, and much more. Some work has been done on detecting these special effects. In compressed domain, however, most work has been focused on gradual transition detection (e.g., fade and dissolve). A review and comparison of some of these techniques can be found in [35].

Zhang et al. [69] use a *twin-comparison* algorithm to detect gradual transitions. A lower threshold (T_t) for frame difference is used to detect potential start of transitions, and a second and larger threshold (T_b) is used for accumulated difference to detect the end of gradual transitions. Either DCT difference metrics [68] or pixel-domain color histogram difference [69] can be used for comparison.

Yeo and Liu [65] use the *absolute difference* between two DC frames as the difference metric. The plot of the difference between the i^{th} and the $(i+k)^{th}$ frames generates a plateau if a gradual transition is present. K is selected to be larger than the length of most gradual transitions.

Meng et al. [37] use the *variance of DCT DC* coefficients in I- and P-frames to detect dissolve. A downward parabolic curve in the variance graph usually indicates a dissolve.

Kobla et al. [31] use features of the DC image sequences (RGB/YUV values or histograms) and apply *FastMap* [17] technique (a dimension reduction algorithm that preserves inter-sample distances) to generate *VideoTrails* [32] of a video sequence in a lower-dimensional space. The trail is segmented and classified into stationary and transitional trails, the latter typically a threaded trail that corresponds to gradual transitions. In [31], VideoTrails and the above three techniques are evaluated by running extensive experiments. The advantage of this technique is that it is good at detecting the beginning and the end of the transition, and can handle a wide variety of editing effects.

One thing that troubles all these techniques is the presence of large object motion and fast camera operations in dynamic scenes. This confuses the gradual transition (e.g., dissolve) detection algorithm and results in high false alarm rates.

3. Sports Video Analysis

Sports video is of wide interest to a large audience. Automatic indexing and analysis of sports video, preferably in the compressed domain, have great potentials in future multimedia applications.

a. Detecting sports video

Sports video [46] usually has characteristic camera motions like long pans (e.g., following a fast break in a basketball game), as well as large magnitudes of motion, high percentage of shots containing significant motion, sudden camera jerks, and frequent appearance of text in score boards, statistics, player's jerseys, etc. These features can be used to detect sports videos.

b. Detection of slow-motion replay in sports video

Slow-motion replay in sports video is usually achieved by slowing the frame rate of the playback, resulting in a single frame to be repeated several times. This causes the presence of still and shift frames. Kobla et al. [30] use information about *macroblock types* in B-frames, along with *vector flow* information, and *number of bits* used for encoding each frame to detect slow-motion replay. Still B-frames usually have a lot of forward-predicted macroblocks, while shift B-frames have a significant amount of backward-predicted macroblocks. Essentially, still B-frames contain very little new information and the number of bits used for coding them is low. On the other hand, the bit rate for the shift frames is higher. Therefore, in a slow-motion replay segment, large variations of bit rates will be observed.

c. Automatic analysis of basketball video

Saur et al. [46] investigate the particular domain of basketball video and use low-level information extracted from MPEG video to detect events like a fast break.

Wide-angle shots are close-up shots are classified based on the motion vector magnitude and number of intra-coded blocks in P-frames. Within wide-angle shots, camera motions are

estimated for panning using the method proposed in [56]. The *accumulated camera motion* is used to detect the fast break event. They also tried to use the irregularities of camera motion to detect steal or a loose ball situation, and use some heuristics and ad hoc methods to detect shooting scenes.

VII. Audio Features and Their Applications

In this section we will discuss low-level audio features that can be extracted from both uncompressed and compressed audio signals, with their applications to audio indexing, analysis, and classification. The term *audio* here refers to generic sound signals, which include speech, dialog, music, songs, radio broadcast, audio tracks of video programs, noise, and mixtures of any of these.

Much of the research work in this area has been focused on indexing and classification of audio clips. Recently, great interests are shown in integrating audio features with video features for better results in video indexing and analysis. The combination of information from both audio and video channels has great potentials in enhancing the power of current video indexing schemes. There is also some interesting research work on music retrieval using tones, notes, MIDI sequences, and structures of music, but this is out of the scope of our discussion in this paper.

1. Audio Features

For the purposes of audio analysis, indexing, and classification, low-level audio features of the sound signals must be first extracted. Although frequency-domain techniques are often

used, most audio feature extraction methods start with uncompressed audio signal in waveforms. The decoding of compressed audio requires much less computation compared with video decoding, and does not pose a computational bottleneck in real applications. Therefore, direct feature extraction from compressed audio, although preferred, is not required in many audio applications.

The following is a list of common audio features. Many audio applications have extracted and used some or all of these features and their variations. Not all of the features can be easily extracted from subband-coded audio signals like MPEG audio. Features that are marked with an asterisk are the ones that can be directly extracted or approximated using subband-coded audio.

a. Time-domain features

1. Short-time energy*
2. Energy/Volume statistics*: mean, standard deviation, dynamic range, etc.
3. Silence ratio*: percentage of low energy audio frames
4. Zero crossing rate (ZCR): number of time-domain zero crossings within a frame
5. Pause rate*: the rate of stop in speech due to separation of words/sentences

b. Frequency-domain (spectral) features

1. Pitch*: fundamental frequency that reveals harmonic properties of audio
2. Subband energy ratio*: histogram-like energy distribution over frequencies
3. Spectral statistics*: centroid, bandwidth, etc.

c. Psycho-acoustic features

1. 4-Hz modulation energy: speech has a characteristic energy modulation peak around the 4Hz syllabic rate.
2. Spectral roll-off point: the 95^{th} percentile of the power spectrum, used to distinguish voiced/unvoiced speech.

2. Audio Applications Using Audio Features

The sub-section discusses some typical applications that use audio features for audio indexing and classification purposes. We use the above labels when citing specific audio features. For example, Feature a.4 represents the zero crossing rate. Note that sometimes a variation of one of the above features is used, which may be slightly different in mathematical form from the original but in essence is the same physically.

a. General audio characterization, classification, and indexing

Patel and Sethi [41] investigate the possibility of extracting audio features directly from MPEG audio and use these features for audio characterization. *Features a.1, a.5, b.1, and b.2* are computed directly using *subband coefficients* of audio signals and are used to classify MPEG audio clips into dialog, non-dialog, and silent categories. Given 81 audio clips from a movie, 96%, 82%, and 100% correct classification rates are reported.

Zhang and Kuo [71] use *Features a.1, a.4, and b.1* to extract low-level audio features for audio retrieval. For coarse classification, using these features alone can help distinguish silence, music, speech, etc. For fine classification, which is more difficult, *Hidden Markov Models* [44] are built for each class of sounds, using these audio features. 90% and 80% classification rates

are reported for coarse and fine classifications, respectively. In [70] these features are used in a *heuristic approach* to segmenting audio data and classifying these segments into silence, speech, songs, music, etc.

b. Speech/Music classification

Scheirer and Slaney [47] collect an array of features (*Features a.3, a.4, b.3, c.1, c.2*, and more) along with their statistics for speech/music classification. They use various *classification frameworks* (multi-dimensional Gaussian MAP estimator, Gaussian Mixture Model, spatial partitioning, and nearest-neighbor classifier) with rigorous training. The best classifier achieves 94% correct classification rate on a frame-by-frame basis.

Saunders [45] uses *statistics of zero crossing rates* (ZCR, Feature a.4) to distinguish speech and music in FM radio programs. The idea is that ZCR exhibits a *bi-model property* (vowels vs. consonants) in speech, but not in music. A 90% classification rate is reported using samples from a 2-hour FM radio program.

Srinivasan et al. [52] use *Features a.1, a.4, and b.1* to segment mixed audio into speech and music based on *heuristic rules*. The task is particularly challenging because of the presence of speech-over-music, fade in and fade out, and special sounds in the audio signal, which is actually common in real-world audio data like the audio track of a TV broadcast. An 80% classification rate is reported, against manually generated ground truth (i.e., a human listener decides if a mixed portion of audio belongs to music or speech segment.).

c. Using audio features for video indexing

Wang et al. [60] use *Features a.2, a.3, and b.3* to characterize different types of video clips associated with the audio tracks. The purpose is to use audio information to help understand the video content (news, weather report, commercials, or football games). Some unique characteristics of the audio features associated with each video type are reported.

Huang et al. [24] use a set of features (*Features a.2, a.3, b.2, b.3, c.1*, and more) to form feature vectors for *audio break detection*. The result is combined with color and motion break detection results to segment videos into scenes.

He et al. [22] use *Feature b.1* to identify the speaker's emphasis in his/her oral presentation, based on the observation that the speaker's introduction of a new topic often corresponds to an increased *pitch range* in his/her voice. This information, combined with slide transition information of the presentation, is used to extract important segments and generate summaries of oral-slide presentations. This interesting application shows the strength of integrating audio features with other types of features, even meta-information.

Naphade et al. [39] used an integrated HMM model (called *multiject*) taking both audio and visual features as observations to detect events such as an explosion. Recently, they also proposed a probabilistic framework using Bayesian networks for semantic-level indexing and retrieval [40]. Sundaram and Chang [53] applied a causal memory based model in detecting audio scenes, which are defined to be segments with long-term consistent audio features. In [54],

they further extended the scene segmentation model to both audio and visual domains and investigated the alignment issues between audio and video scenes.

So far the research work in audio indexing and classification is still in the early stage. There is much more to explore in this area, not only new features and feature extraction methods (from both compressed and uncompressed audio), but also high-level methodologies that have been widely and successfully applied to speech recognition and speaker identification fields, such as Hidden Markov Models, Gaussian Mixture Models, classification frameworks, etc. Another important issue that requires systematic investigation is how to seamlessly integrate audio features with visual, textual, and meta-information for indexing and analysis tasks in multimedia applications.

VIII. Summary and Evaluation of Video and Audio Features

In this section we will summarize the video and audio features that we have surveyed in previous sections. We do not intend to exhaustively list all the features that we have mentioned. Instead, we will only select a *subset* of these features that we believe are effective, fairly robust, and useful in key applications. We will evaluate these features in the following aspects: *efficiency, effectiveness, and limitations*. Efficiency refers to the computational requirement, e.g., the number of frames that the feature extraction process can be done per second. Effectiveness refers to the usefulness of the feature in practical applications. Limitation refers to the constraints caused by practical implementations, such as the block resolution limit imposed by MPEG compression. Note that it is impossible for us to implement and test all these features. Therefore, the evaluation is based on our own experience with some of these features, our colleagues'

experience, reviews, and discussion. Although often qualitative in nature, the evaluation aims at presenting a compact summary of important techniques in this research field.

The features will be grouped in spatial visual, motion, coding, and audio features. We will also briefly review the visual descriptors, audio descriptors, and multimedia description schemes in the current MPEG-7 standardization process, and discuss the possibility of extracting some of theses descriptors from MPEG-1/2 compressed video and audio streams.

Note that when discussing the efficiency of feature extractions, we assume that the computer being used is an average new PC, e.g., a PC with a 500 MHz Pentium III processor and 128 MB of memory. We roughly describe the efficiency of each feature extraction process by *very efficient* (more than 30 frames/s), *efficient* (10-30 frames/s), or *less efficient* (less than 10 frame/s). For the effectiveness of the features, we use *highly effective* to refer to robust features that are important in most key applications, and *moderately effective* to refer to less robust features that are useful for specific applications in limited domains.

1. Spatial Visual Features

DCT DC image sequence is highly effective in capturing the global, iconic view of a video sequence. The construction of DCT DC sequences is very efficient using approximated DC values for P- and B-frames and can be done in real time (30 frames/s or faster). The limitation is that the final several P- and B-frames in a GOP may suffer severe quality degradation due to error accumulation during successive approximations.

a. Color

- *DC color histogram* – very efficient to extract, highly effective global feature for frames, but resolution limited for small regions or objects.
- *DC YCbCr vector* – The elements of this vector are the average Y, Cb, and Cr values of all the macroblocks in a frame. It is very efficient to extract, but needs expensive dimension reduction technique (e.g., the method used in VideoTrails [32]) to make it a highly effective representation of the color feature of the video frame.

b. Texture and edge

- *DCT AC coefficient energy* – very efficient to extract, moderately effective for the approximation of texture, but may be useful in specific domains only.
- *DCT block edge map* – efficient to extract, highly effective approximation of edge activities, but difficult to link edge segments to form long, smooth edges.

2. Motion Features

a. Motion vector field

- *Motion vector based frame segmentation* – efficient to extract, moderately effective for the detection of large moving regions, but sensitive to motion vector errors.

b. Camera operation estimation

- *3-D coordinate, 6-parameter model* – less efficient to extract, highly effective for the estimation of most camera operations, but sensitive to motion vector errors, particularly in initial estimation.

- *3-D coordinate, 3-parameter model* – efficient to extract, highly effective for the estimation of dominant camera operations, limited to pan, tilt, and zoom operations.
- *2-D affine model* – efficient to extract, effective for the detection of pan and zoom operations, but sensitive to motion vector errors.

c. Motion vector statistics

- *Motion activity descriptor* – very efficient to extract, moderately effective for rough characterization of frame motion properties, but with limited discrimination capability.
- *Motion magnitude and directional histogram* – very efficient to extract, moderately effective for local motion characterization, but with limited accuracy due to motion vector errors and the sparseness of motion vectors.

3. Video Coding Features

- *Macroblock type* – very efficient to extract, moderately effective for shot boundary detection in inter-coded frames, performance affected by encoder implementation, better and more robust results can be achieved by combining color features.

4. Audio Features

Because the amount of audio data is much smaller than video data, and decoding of compressed audio is much cheaper than that of video, almost all of the audio features can be highly efficiently extracted, either from compressed audio, or from uncompressed audio waveform. Therefore, in the following evaluation of the complexity of these features, we will only indicate whether the feature can be directly extracted or approximated from compressed audio.

- *Short-time energy* – can be extracted from compressed audio. It indicates the average loudness of the audio signal in short periods, and is highly effective for silence detection.
- *Energy statistics* – can be extracted from compressed audio, moderately effective for audio classification, typically combined with other audio features.
- *Silence ratio* – can be extracted from compressed audio, moderately effective for audio classification, typically combined with other audio features.
- *Zero crossing rate* – cannot be extracted from compressed audio, highly effective for the differentiation of speech and music type of audio.
- *Pitch* – can be approximated using compressed audio, highly effective for the detection of harmonic properties of audio, but difficult to estimate for noisy audio and some speech segments.
- *Spectral statistics* – can be approximated using compressed audio, moderately effective for music/speech classification and general audio classification, typically combined with other audio features.

It can be seen from the above that audio indexing and classification usually use an array of audio features as the foundation for further analysis. A single feature or two are not enough. Among all these audio features, short-time energy, zero crossing rate, and pitch are the most popular ones.

5. Brief Review of MPEG-7 Descriptors

MPEG-7 is an ISO/IEC international standard currently under development, which is expected to reach the International Standard (IS) stage in the fall of 2001. Formally known as *Multimedia Content Description Interface*, it will standardize a set of *descriptors* (Ds), a set of

description schemes (DSs), a language to specify description schemes (and possibly descriptors), i.e., *the Description Definition Language (DDL)*, and one or more ways to encode descriptions. A descriptor is a representation of a feature, while a description scheme specifies the structure and semantics of the relationships between its components, which may be both Ds and DSs. With these tools, MPEG-7 aims at creating a standard for the description of multimedia content that is useful for a wide range of applications, such as multimedia digital libraries, broadcast media selection, multimedia editing, home entertainment devices, and so forth. More information about MPEG-7 can be found at the MPEG website http://drogo.cselt.it/mpeg/.

Universal access to a great wealth of multimedia information demands tools and techniques that help users quickly get the content they need. This is clearly a driving force behind MPEG-7. Active research on content-based image/video indexing and retrieval also has a direct impact on the formation of MPEG-7. However, MPEG-7 does not standardize how audiovisual features are extracted, automatically or manually. Nor does it standardize how these descriptions are used for search and retrieval. These interesting and often open questions are left to the innovations of research communities and industries.

In this subsection, we will briefly review important MPEG-7 visual and audio Ds. Particularly, we will discuss the possibility to extract features and build these Ds directly from MPEG-1/2/4 compressed data, as feature extraction is beyond the scope of MPEG-7. Looking at compressed-domain feature extraction from the MPEG-7 perspective is valuable because the majority of multimedia contents that MPEG-7 will describe are already in compressed formats like MPEG-1, MPEG-2, and MPEG-4. At the La Baule meeting in October 2000, MPEG-7

reached the stage of Committee Draft (CD), which is sufficiently stable. Therefore, although MPEG-7 standardization is still ongoing, we will use documents from the La Baule meeting (ISO/IEC JTC1/SC29/WG11 N3673, W3703, N3704, N3751, N3752) for the review. The MPEG-7 standard consists of seven parts, namely, *systems*, *description definition language*, *audio*, *visual*, *multimedia description schemes*, *reference software*, and *conformance*. We will focus on MPEG-7 standard parts that are relevant to audio-visual feature extraction, i.e., parts 3 (audio) and 4 (visual). We will not try to review the list of features exhaustively, but only highlight those important audio and visual descriptors.

a. MPEG-7 visual

MPEG-7 visual description tools consist of basic structures and visual descriptors for color, texture, shape, motion, localization, and others.

Basic structures

- *Grid layout* – This is a splitting of the image into multiple uniform-sized rectangular regions, so that each region can be described separately. In the compressed domain, since only the DCT DC values are readily available, the dimension of the rectangular regions is usually limited to integer multiples of eight pixels (i.e., DCT block dimension).
- *Time series* – This descriptor defines a temporal series of descriptors in a video segment. In the compressed domain, using a temporal series of features extracted from the DCT DC image sequence is effective for video segmentation and matching purposes.

Color descriptors

- *Color space* – This information is available in the MPEG-1/2 video stream (YCbCr triplets). Transformations to other color spaces are simple and efficient.
- *Dominant colors* – This can be extracted using the DC image for either the whole image (e.g., flag or color trademark images) or large image regions within the frame.
- *Scalable color* – This descriptor is a color histogram in HSV color space, which is encoded by a Haar transform. In the compressed domain, this can be computed using DC image. Accuracy of histogram may be degraded due to the small number of samples, where color histogram of a group of frames (GoF/GoP color) can be used instead.
- *GoF/GoP color* – This is the extension of the scalable color descriptor to a video segment or a collection of still images. It may be preferred in the compressed domain because color samples from multiple frames are accumulated for more robust computation of the histogram, assuming that colors vary slowly within a video segment. Average or median histogram can be used.
- *Color layout* – This descriptor specifies spatial distribution of colors for high-speed retrieval or browsing. An 8x8 matrix of dominant local colors is DCT transformed, quantized, and zigzag scanned. In the compressed domain, the DCT DC image can be segmented into an 8x8 grid layout and used to generate the matrix of dominant colors.
- *Color structure* – Color structure information is embedded in this descriptor by taking into account colors of neighboring pixels, instead of considering each pixel separately as in color histogram. Its main purpose is still-image matching for similarity-based image retrieval. In the compressed domain, the DCT DC image is a subsampled version so that

the 8x8 structuring element actually covers 64x64 regions in the original frame. Therefore, color structures in local neighborhoods are not preserved, and the descriptor may not have clear advantages over normal color histogram.

Texture descriptors

- *Homogeneous texture* – The extraction of this descriptor involves filtering the image with filter banks and calculating moments in corresponding subbands. It is therefore difficult to extract directly in the compressed domain. However, it can be used to describe full-resolution images of key video frames.
- *Texture browsing* – This descriptor characterizes texture in terms of regularity, coarseness, and directionality. The extraction is similar to the above, and is also difficult in the compressed domain.
- *Edge histogram* – This descriptor represents the spatial distribution of different types of edges. It is difficult to extract directly in the compressed domain. Due to low resolution of the DC images and block effects, it is very hard to extract reliable edges from compressed video.

Shape descriptors

- *Region-based shape* – This descriptor uses a set of Angular Radial Transform (ART) coefficients to represent complex shapes. In the compressed domain, color and motion information can be used to segment video frames. This descriptor can then be computed approximately using the segmented regions in the DC image. Obviously, region-based shape is more reliable for large objects or regions in compressed video.

- *Contour-based shape* – This descriptor uses the Curvature Scale-Space representation of the contour. It is difficult to extract directly in the compressed domain due to the hard problems of reliable edge detection and contour extraction from compressed video.

Motion descriptors

- *Camera motion* – This descriptor supports an array of basic camera operations, and captures the temporal and visual attributes of camera motion of sub-shots, either in single or mixed mode. As we have discussed in previous sections, techniques are available for estimating camera motion using motion vectors in MPEG video, such as [6, 36, 57]. Some of them are able to estimate the global camera motion when there are multiple types of basic camera motion simultaneously (mixed mode), while others work better when there is only one dominant type of camera motion.
- *Motion trajectory* – This descriptor is a list of key points in 2-D or 3-D Cartesian coordinates, along with optional interpolating functions. For compressed video, it is possible to generate motion trajectories of dominant objects or regions by estimating region motion using MPEG motion vectors [2, 12], although spurious motion vectors may degrade the accuracy of the trajectories.
- *Parametric motion* – This descriptor characterizes the evolution of arbitrarily shaped regions over time in terms of a 2-D geometric transform, such as the affine model, the planer perspective model, and the quadratic model. It addresses both region motion and global motion. In the compressed domain, parametric motion can be estimated using motion vectors to approximate optical flow and adopting a 2-D geometric transform, such as the affine model [6, 36].

- *Motion activity* – This descriptor include motion attributes of video segments such as intensity of activity, direction of activity (if any), spatial distribution of activity, and spatial/temporal localization of activity. As shown in [14], this descriptor can be computed efficiently using motion vectors in MPEG compressed video.

b. MPEG-7 Audio

Although less mature than MPEG-7 visual, MPEG-7 audio has reached the fairly stable stage of Committee Draft in the October 2000 MPEG-7 meeting at La Baule. MPEG-7 audio tools can be put in two general categories: low-level audio description, and application-driven description. The low-level description tools are applicable to general audio data, regardless of the specific content (e.g., song or speech) carried by the audio signal. The application-driven description tools are applicable to specific types of audio content, such as speech, sound effects, musical instruments, and melodies.

Low-level description tools

- *Scalable series* – These are abstract data types for series of scalar or vector values, which allow temporal series of audio descriptors to be represented in a scalable fashion. These are one of the foundations of low-level description tools.
- *Audio description framework* – This is a collection of low-level audio descriptors (features), which include *waveform envelope*, *spectrum envelope*, *audio power*, *spectrum centroid*, *spectrum spread*, *fundamental frequency*, and *harmonicity*. Many of these descriptors are similar or identical to the audio features we discussed in Section VII. In the MPEG compressed audio (where the signal is subband coded), extractions of some of these low-level descriptors like waveform envelope and audio power are straightforward.

For other descriptors, such as spectrum envelope, centroid, spread, and fundamental frequency (pitch), subband coefficients can be used directly for estimation, without the need to fully decode the MPEG audio [24, 41, 60].

- *Silence* – Silence descriptor describes silent sound segments. This can be used for audio segmentation, or semantic-level event detection when combined with other audiovisual descriptors. Silence can be detected in MPEG compressed audio directly [41].

Application-driven description tools

- *Spoken content description tools* – These consist of a number of combined word and phoneme lattices in an audio stream. By combining the lattices, the problem of out-of-vocabulary words in speech recognition and retrieval is greatly alleviated. Possible applications are indexing and retrieval of audio stream, and indexing and retrieval of multimedia objects annotated with speech.
- *Timbre description tools* – These tools aims at describing the perceptual features (*timbre*) of musical instruments that distinguish one from another even when the sounds produced have the same pitch and loudness. A reduced set of descriptors is selected for this purpose, such as log-attack time, harmonic centroid, spread, and deviation, etc. Possible applications are authoring tools for sound engineers and musicians, and retrieval tools for producers.
- *Sound effects description tools* – These are a collection of tools for the indexing and categorization of general sound effects. Potential applications include automatic segmentation and indexing sound tracks.

- *Melody contour description tools* – These tools are a compact representation of melodies that allows for efficient and robust melody-similarity matching. A typical application is query by humming, which is useful in music search and retrieval.

Although some low-level audio features can be estimated directly using compressed audio signal (such as MPEG-1 audio), most of the application-driven audio description tools mentioned above cannot be applied directly in the compressed domain. However, since audio signal is one-dimensional and the data volume is much smaller compared with video, it is often affordable to fully decompress the audio signal before further analysis is done.

IX. Conclusion and Future Directions

In this paper we survey video and audio features that can be extracted or approximated using compressed video and audio like MPEG-1/2 streams. Without expensive decompressions, features can be efficiently extracted for large archives of multimedia data. However, compressed-domain features have their limitations, which are reflected in the survey, due to the lower resolution, block effects, inaccurate motion vectors, etc.

We investigate spatial visual features, motion features, video coding features, and audio features, along with their potential applications. Some important applications using multiple visual and motion features are discussed in a separate section. Finally, we summarize and evaluate a small set of features that we believe are important and fairly robust. We also review the recent status of MPEG-7 visual descriptors, and audio descriptors, and discuss the possibility of extracting MPEG-7 features directly from compressed video and audio. We believe that

compressed-domain feature extraction will be valuable to MPEG-7 applications with the availability of the vast amount of MPEG-1/2 content.

Comparing the MPEG-7 visual descriptors with the features that we have surveyed, it becomes clear that for a feature to be successful and be able to become part of the standard, it has to be *robust*, *general*, and *cost-effective*. Features whose applications are limited to a small domain typically are not included in standards.

Features alone are far from enough for video/audio indexing and analysis. Although feature extractions (especially from the compressed domain) will continue to be a research topic, equally important problems are (1) how to obtain semantic-level knowledge and understanding from these low-level features, and (2) the synergic integration of features from multiple media such as video, audio, text, and metadata. These problems are important, challenging, and demand rigorous and systematic investigation.

X. References

1. A. Akutsu, Y. Tonomura, H. Hashimoto, and Y. Ohba, "Video indexing using motion vectors," *Proc. SPIE Conference on Visual Communications and Image Processing*, SPIE Vol. 1818, pp. 1522-1530, 1992.
2. E. Ardizzone, M. La Cascia, A. Avanzato, and A. Bruna, "Video indexing using MPEG motion compensation vectors," *Proc. IEEE International Conference on Multimedia Computing and Systems*, 1999.

3. E. Ardizzone, M. La Cascia, and D. Molinelli, “Motion and color based video indexing and retrieval,” *Proc. International Conference on Pattern Recognition*, 1996.
4. F. Arman, A. Hsu, and M.-Y. Chiu, “Image processing on compressed data for large video databases,” *Proc. ACM Multimedia 93*, pp. 267-272, 1993, Anaheim, CA.
5. O. K.-C. Bao, J. A. Lay, and L. Guan, “Compressed domain video parsing using energy histograms of the lower frequency DCT coefficients,” *Proc. SPIE Conference on Storage and Retrieval for Multimedia Database 2000*, pp. 293-300, Jan. 2000, San Jose, CA.
6. J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani, “Hierarchical model-based motion estimation,” *Proc. 2nd European Conference on Computer Vision*, 1992.
7. G. Boccignone, M. De Santo, and G. Percannella, “An algorithm for video cut detection in MPEG sequences,” *Proc. SPIE Conference on Storage and Retrieval of Media Databases 2000*, pp. 523-530, Jan. 2000, San Jose, CA.
8. J. S. Boreczky and L. A. Rowe, “Comparison of video shot boundary detection techniques,” *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases IV*, pp. 170-179, Jan. 1996.
9. S.-F. Chang and D. G. Messerschmit, “Manipulation and compositing of MC-DCT compressed video”, *IEEE Journal on Selected Areas in Communications, Special Issue on Intelligent Signal Processing*, Jan. 1995, pp. 1-11.
10. J.-Y. Chen, C. Taskiran, E. J. Delp, and C. A. Bouman, “ViBE: a new paradigm for video database browsing and search,” *Proc. IEEE Workshop on Content-Based Access of Image and Video Databases*, 1998.
11. J. W. Davis, “Recognizing movement using motion histograms,” Technical Report No. 487, MIT Media Laboratory Perceptual Computing Section, April 1998.

12. N. Dimitrova and F. Golshani, “Motion recovery for video content analysis,” *ACM Trans. Information Systems*, Vol. 13, No. 4, October 1995, pp. 408-439.
13. A. Divakaran, H. Ito, H. Sun, and T. Poon, “Scene change detection and feature extraction for MPEG-4 sequences,” *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases VII*, pp. 545-551, Jan. 1999, San Jose, CA.
14. A. Divakaran and H. Sun, “A descriptor for spatial distribution of motion activity for compressed video,” *Proc. SPIE Conference on Storage and Retrieval for Media Databases 2000*, pp. 392-398, Jan. 2000, San Jose, CA.
15. Dolby Laboratories, Dolby AC-3: Multi-channel perceptual coding (http://www.dolby.com/tech/multipc.html), 1997.
16. H.-L. Eng and K.-K. Ma, “Motion trajectory extraction based on macroblock motion vectors for video indexing,” *Proc. IEEE International Conference on Image Processing*, Oct. 1999, Kobe, Japan.
17. C. Faloutsos and K. Lin, “FastMap: a fast algorithm for indexing, data mining, and visualization of traditional and multimedia databases,” *Proc. ACM SIGMOD Conference*, pp. 163-174, 1995.
18. J. Feng, K. T. Lo, and H. Mehrpour, “Scene change detection algorithm for MPEG video sequence,” *Proc. IEEE International Conference on Image Processing*, 1996, Lausanne, Switzerland.
19. W. A. C. Fernando, C. N. Canagarajah, and D. R. Bull, “Video segmentation and classification for content based storage and retrieval using motion vectors,” *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases VII*, pp. 687-698, Jan. 1999, San Jose, CA.

20. R. M. Ford, “A quantitative comparison of shot boundary detection metrics,” *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases VII*, pp. 666-676, Jan. 1999, San Jose, CA.
21. B. G. Haskell, A. Puri, and A. N. Netravali, *Digital Video: An Introduction to MPEG-2*, Chapman and Hall, 1997.
22. L. He, E. Sanocki, A. Gupta, and J. Grudin, “Auto-summarization of audio-video presentations,” *Proc. ACM Multimedia 99*, pp. 489-498, Oct. 1999, Orlando, FL.
23. Y. S. Ho and A. Gersho, “Classified transform coding of images using vector quantization,” *Proc. ICASSP 89*, pp. 1890-1893, 1989.
24. J. Huang, Z. Liu, and Y. Wang, “Integration of audio and visual information for content-based video segmentation,” *Proc. IEEE International Conference on Image Processing*, 1998.
25. ISO/IEC IS 11172 – 2, MPEG-1 Video.
26. ISO/IEC IS 11172 – 3, MPEG-1 Audio.
27. ISO/IEC IS 13818 – 2, MPEG-2 Video.
28. ISO/IEC IS 13818 – 3, MPEG-2 Audio BC.
29. ISO/IEC IS 13818 – 7, MPEG-2 Audio AAC.
30. V. Kobla, D. DeMenthon, and D. Doermann, “Detection of slow-motion replay sequences for identifying sports videos,” *Proc. IEEE Workshop on Multimedia Signal Processing*, 1999.
31. V. Kobla, D. DeMenthon, and D. Doermann, “Special effect edit detection using VideoTrails: a comparison with existing techniques,” *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases VII*, 1999.

32. V. Kobla, D. Doermann, and C. Faloutsos, “VideoTrails: representing and visualizing structure in video sequences,” *Proc. ACM Multimedia 97*, pp. 335-346, 1997.
33. V. Kobla, D. Doermann, and K.-I. Lin, “Archiving, indexing, and retrieval of video in the compressed domain,” *Proc. SPIE Conference on Multimedia Storage and Archiving Systems*, SPIE Vol. 2916, pp. 78-89, 1996.
34. V. Kobla, D. Doermann, K.-I. Lin, and C. Faloutsous, “Compressed domain video indexing techniques using DCT and motion vector information in MPEG video,” *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases V*, SPIE Vol. 3022, pp. 200-211, 1997.
35. R. Lienhart, “Comparison of automatic shout boundary detection algorithms,” *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases VII*, pp. 290-301, Jan. 1999, San Jose, CA.
36. J. Meng and S.-F. Chang, “CVEPS – a compressed video editing and parsing system,” *Proc. ACM Multimedia 96*, 1996.
37. J. Meng, Y. Juan, and S.-F. Chang, “Scene change detection in an MPEG compressed video sequence,” *IS&T/SPIE Symposium Proceedings*, Vol. 2419, Feb. 1995, San Jose, California.
38. J. Nang, S. Hong, and Y. Ihm, “An efficient video segmentations scheme for MPEG video stream using macroblock information,” *Proc. ACM Multimedia 99*, pp. 23-26, Oct. 1999, Orlando, FL.
39. M. R. Naphade, T. Kristjansson, B.J. Frey, and T.S. Huang, “Probabilistic multimedia objects (Multijects): a novel approach to video indexing and retrieval in multimedia

systems," *Proc. IEEE International Conference on Image Processing*, Vol. 5, Oct. 1998, Chicago, IL.

40. M. R. Naphade and T. S. Huang, "A probabilistic framework for semantic indexing and retrieval in video," *Proc. IEEE International Conference on Multimedia and Expo*, Vol. 1, Jul.-Aug. 2000, New York, NY.
41. N. V. Patel and I. K. Sethi, "Audio characterization for video indexing," *Proc. SPIE Conference on Storage & Retrieval for Image and Video Databases IV*, Feb. 95, San Jose, CA.
42. N. V. Patel and I. K. Sethi, "Video shot detection and characterization for video databases," *Patter Recognition*, Vol. 30, No. 4, pp. 583-592, 1997.
43. A. J. Patti, M. I. Sezan, and A. M. Tekalp, "Robust methods for high quality stills from interlaced video in the presence of dominant motion," *IEEE Trans. Circuits and Systems for Video Technology*, Vol. 7, No. 2, pp. 328-342, April 1997.
44. L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Prentice Hall, 1993.
45. J. Saunders, "Real-time discrimination of broadcast speech/music," *Proc. ICASSP 96*, pp. 993-996, 1996.
46. D. D. Saur, Y.-P. Tan, S. R. Kulkarni, and P. J. Ramadge, "Automated analysis and annotation of basketball video," *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases V*, SPIE Vol. 3022, pp. 176-187, 1997.
47. E. Scheirer and M. Slaney, "Construction and evaluation of a robust multi-feature speech/music discriminator," *Proc. ICASSP 97*, 1997.

48. B. Shen and I. K. Sethi, “Convolution-based edge detection for image/video in block DCT domain,” *Journal of Visual Communications and Image Representation*, Vol. 7, No. 4, pp. 411-423, Dec. 1996.
49. B. Shen and I. K. Sethi, “Direct feature extraction from compressed images,” *Proc. SPIE Conference on Storage and Retrieval for Image and Video Database IV*, Jan. 1996.
50. K. Shen and E. Delp, “A fast algorithm for video parsing using MPEG compressed sequences,” *Proc. IEEE International Conference on Image Processing*, Vol. 2, pp. 252-255, 1995.
51. B. C. Song and J. B. Ra, “Fast edge map extraction from MPEG compressed video data for video parsing,” *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases VII*, pp. 710-721, Jan. 1999, San Jose, CA.
52. S. Srinivasan, D. Petkovic, and D. Ponceleon, “Towards robust features for classifying audio in the CueVideo system,” *Proc. ACM Multimedia 99*, pp. 393-400, Oct. 1999, Orlando, FL.
53. H. Sundaram and S.-F. Chang, “Audio scene segmentation using multiple models, features and time scales,” *Proc. ICASSP 2000*, Istanbul, Turkey, Jun. 2000.
54. H. Sundaram and S.-F. Chang, “Determining computable scenes in films and their structures using audio-visual memory models,” *Proc. ACM Multimedia 2000*, Oct.-Nov. 2000, Los Angeles, CA.
55. Y.-P. Tan, S. R. Kulkarni, and P. J. Ramadge, “A framework for measuring video similarity and its application to video query by example,” *Proc. IEEE International Conference on Image Processing*, Oct. 1999, Kobe, Japan.

56. Y.-P. Tan, S. R. Kulkarni, and P. J. Ramadge, “A new method for camera motion parameter estimation,” *Proc. IEEE International Conference on Image Processing*, Vol. 1, pp. 406-409, 1995.

57. Y.-P. Tan, D. D. Saur, S. R. Kulkarni, and P. J. Ramadge, “Rapid estimation of camera motion from compressed video with application to video annotation,” *IEEE Trans. on Circuits and Systems for Video Technology*, 1999.

58. Y. Taniguchi, A. Akutsu, and Y. Tonomura, “PanoramaExcerpts: extracting and packing panoramas for video browsing,” *Proc. ACM Multimedia 97*, pp. 427-436, 1997.

59. Y. T. Tse and R. L. Baker, “Camera zoom/pan estimation and compensation fore video compression,” *Proc. SPIE Conf. On Image Processing Algorithms and Techniques II*, Boston, MA, pp. 468-479, 1991.

60. Y. Wang, J. Huang, Z. Liu, and T. Chen, “Multimedia content classification using motion and audio information,” *Proc. ISCAS 97*, Vol. 2, pp. 1488-1491, Hong Kong, June 1997.

61. W. Wolf, “Key frame selection by motion analysis,” *Proc. ICASSP 96*, Vol. II, pp. 1228-1231, 1996.

62. C. S. Won, D. K. Park, I. Y. Na, and S.-J. Yoo, “Efficient color feature extraction in compressed video,” *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases VII*, pp. 677-686, Jan. 1999, San Jose, CA.

63. B.-L. Yeo and B. Liu, “On the extraction of DC sequence from MPEG video,” *Proc. IEEE International Conference on Image Processing*, Vol. 2, pp. 260-263, 1995.

64. B.-L. Yeo and B. Liu, “Rapid scene analysis on compressed videos,” *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 5, No. 6, pp. 533-544, 1995.

65. B.-L. Yeo and B. Liu, “Unified approach to temporal segmentation of Motion JPEG and MPEG video,” *Proc. International Conference on Multimedia Computing and systems,* pp. 2-13, 1995.

66. T. Yoshida and Y. Sakai, “Reliability metric of motion vectors and its application to motion estimation,” *Proc. SPIE Conference on Visual Communications and Image Processing 95*, pp. 799-809, 1995.

67. R. Zabih, J. Miller, and K. Mai, “A feature-based algorithm for detecting and classifying scene breaks,” *Proc. ACM Multimedia 95*, pp. 189-200, Nov. 1995, San Francisco, CA.

68. H. J. Zhang, C. Y. Low, and S. W. Smoliar, “Video parsing and browsing using compressed data,” *Multimedia Tools and Applications,* Vol.1, No. 1, pp. 89-111, 1995.

69. H. J. Zhang, A. Kankanhalli, and S. W. Smoliar, “Automatic partitioning of full-motion video,” *Multimedia Systems Journal*, Vol. 1, No. 1, pp. 10-28, 1993.

70. T. Zhang and C.-C. J. Kuo, “Heuristic approach for generic audio data segmentation and annotation,” *Proc. ACM Multimedia 99*, pp. 67-76, Oct. 1999, Orlando, FL.

71. T. Zhang and C.-C. J. Kuo, “Hierarchical classification of audio data for archiving and retrieving,” *Proc. ICASSP 99*, Vol. 6, pp. 3001-3004, 1999.

72. Y. Zhong, H. J. Zhang, and A. K. Jain, “Automatic caption localization in compressed video,” *Proc. IEEE International Conference on Image Processing*, Oct. 1999, Kobe, Japen.

List of Figures

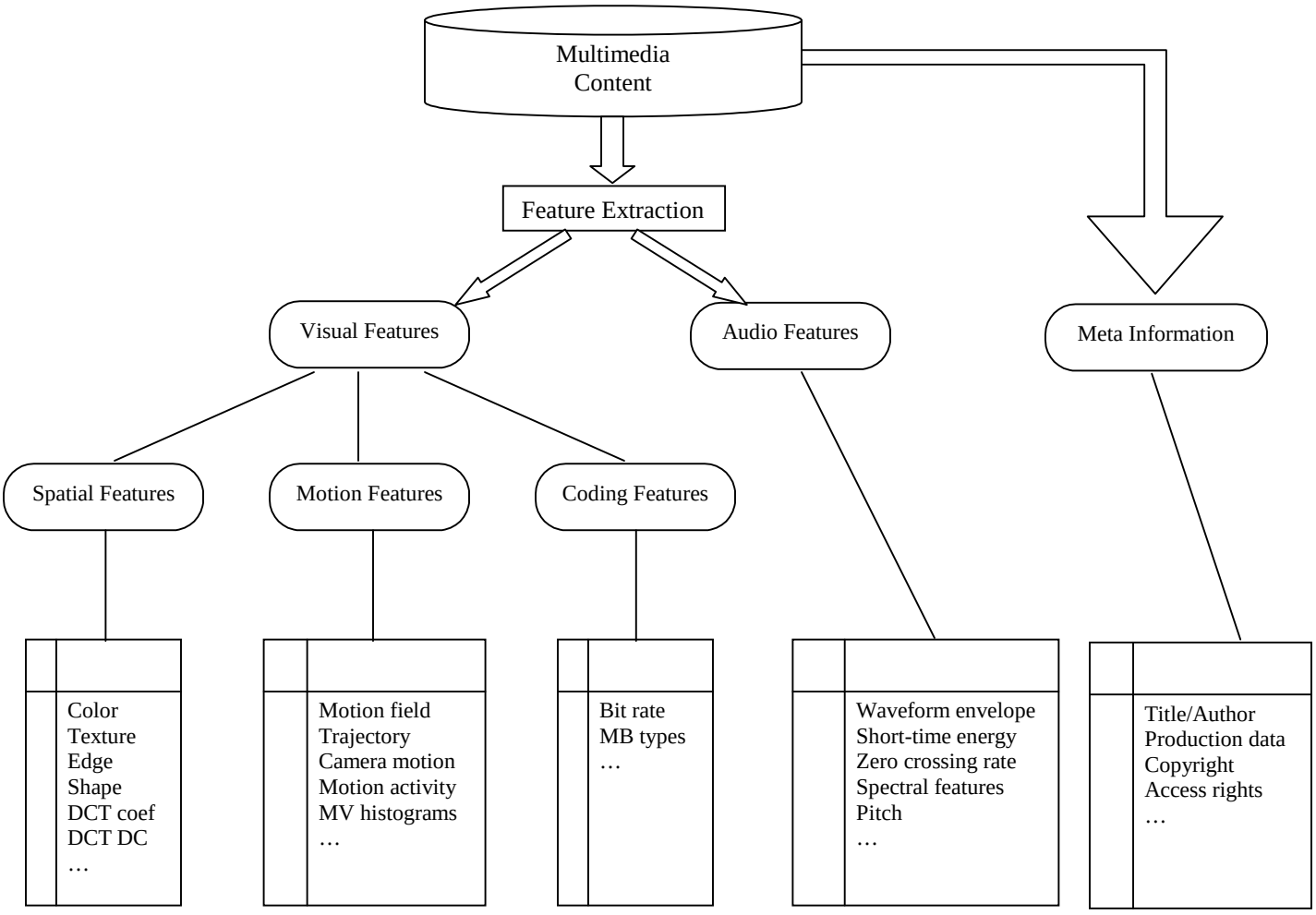

Figure 1: Hierarchy of commonly used features in video/audio indexing and analysis.

Category	Feature	Efficiency	Effectiveness	Limitations	Section
COLOR	*DC color histogram*	Very efficient	High	Degraded for small regions	III.2
	DC YCbCr vector	Very efficient	High	Needs expensive dimension reduction techniques	III.2
TEXTURE & EDGE	*DCT AC coefficient energy*	Very efficient	Moderate	Coarse estimate of texture	III.3
	DCT block edge map	Efficient	High	Difficult to link edge segments	III.3
CAMERA OPERATION	*3-D, 6-patameter model*	Less efficient	High	Sensitive to motion vector errors in initial estimation	IV.2
	Affine, 3-paramter model	Efficient	High	Pan, tilt, and zoom only	IV.2
MOTION STATISTICS	*Motion activity*	Very efficient	Moderate	Rough characterization of motion properties	IV.3
	Motion histograms	Very efficient	Moderate	Limited by motion vector errors and sparseness	IV.3
CODING PARAMETERS	*Macroblock type information*	Very efficient	Moderate	Affected by encoder implementation	V.1
AUDIO FEATURES	*Short-time energy*	Very efficient	High	Coarse representation of the waveform	VII.1
	Energy statistics	Very efficient	Moderate	Combined with other features for classification	VII.1
	Silence ratio	Very efficient	Moderate	Combined with other features for classification	VII.1
	Zero crossing rate	Very efficient	High	Difficult to extract directly in compressed audio	VII.1
	Pitch	Very efficient	High	Difficult to estimate for noisy audio	VII.1
	Spectral statistics	Very efficient	Moderate	Combined with other features for classification	VII.1

Figure 2: Summary of a subset of effective and fairly robust features.

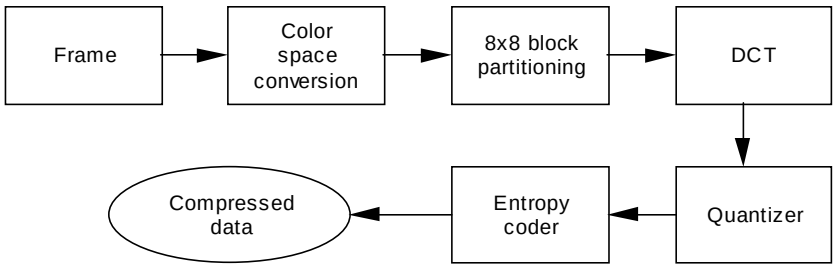

Figure 3: Block diagram of a typical encoding sequence using the DCT.

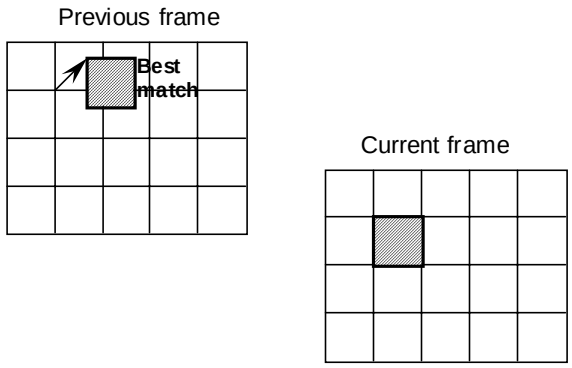

Figure 4: Block-based motion compensation.

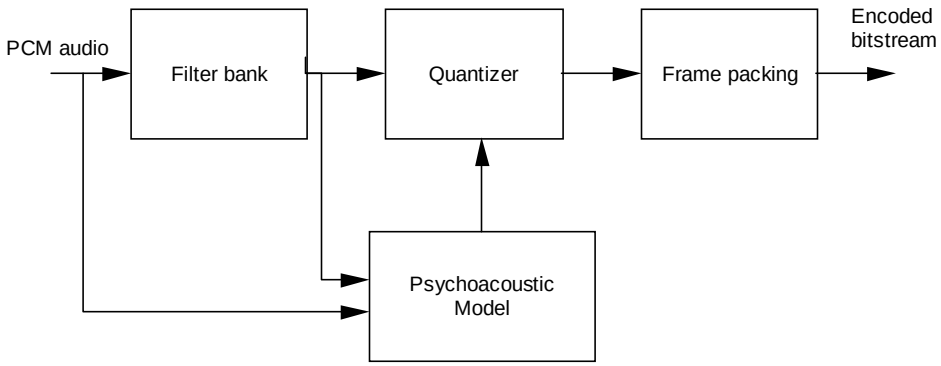

Figure 5: Basic structure of an MPEG-1 audio encoder.