Video Data Mining: Semantic Indexing and Event Detection from the Association Perspective

Xingquan Zhu, *Member*, *IEEE*, Xindong Wu, *Senior Member*, *IEEE*,
Ahmed K. Elmagarmid, *Senior Member*, *IEEE*, Zhe Feng, and Lide Wu, *Senior Member*, *IEEE*

Abstract—Advances in the media and entertainment industries, including streaming audio and digital TV, present new challenges for managing and accessing large audio-visual collections. Current content management systems support retrieval using low-level features, such as motion, color, and texture. However, low-level features often have little meaning for naive users, who much prefer to identify content using high-level semantics or concepts. This creates a gap between systems and their users that must be bridged for these systems to be used effectively. To this end, in this paper, we first present a knowledge-based video indexing and content management framework for domain specific videos (using basketball video as an example). We will provide a solution to explore video knowledge by mining associations from video data. The explicit definitions and evaluation measures (e.g., temporal support and confidence) for video associations are proposed by integrating the distinct feature of video data. Our approach uses video processing techniques to find visual and audio cues (e.g., court field, camera motion activities, and applause), introduces multilevel sequential association mining to explore associations among the audio and visual cues, classifies the associations by assigning each of them with a class label, and uses their appearances in the video to construct video indices. Our experimental results demonstrate the performance of the proposed approach.

Index Terms—Video mining, multimedia systems, database management, knowledge-based systems.

1 Introduction

Organizations with large digital assets have a need to retrieve meaningful information from their digital collections. Applications such as digital libraries, video-on-demand systems, and interactive video applications introduce new challenges in managing large collections of audio-visual content. To help users find and retrieve relevant video more effectively and to facilitate new and better ways of entertainment, advanced technologies must be developed for indexing, filtering, searching, and mining the vast amount of videos. Motivated by these demands, many video research efforts have been made on exploring more efficient content management systems. A simple framework is to partition continuous video frames into discrete physical shots and extract low-level features from video shots to support activities like searching, indexing [42], [43], or retrieval [1]. Unfortunately, a single shot which is separated from its context has less capability of conveying semantics. Moreover, the index considering only visual similarities ignores the temporal information among shots. Consequently, the constructed cluster nodes may contain shots that have considerable variances both in semantics and visual content and, therefore, do not make much sense to human perception. The solution to this problem is to explore video knowledge to construct a database indexing structure which can facilitate database management and access. However, despite the fact that video was invented for more than 50 years and has been widely accepted as an excellent and popular tool to represent information, one can find that it has never been an easy operation to extract or explore knowledge from video data [2], [3], [4], [5].

Recently, there has been a trend of employing various data mining techniques [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19] in exploring knowledge from large video sets. These efforts are motivated by successful data mining algorithms and by the tremendous appeal of efficient video database management. Consequently, many video mining approaches have been proposed, which can be roughly classified into three categories:

1. Special pattern detection [6], [7], [8], [9], [16], [17], [18], which detects special patterns that have been modeled in advance, and these patterns are usually characterized as video events (e.g., dialog, or presentation).
2. Video clustering and classification [10], [11], [12], [15], [19], which clusters and classifies video units into different categories. For example, in [10], [11], video clips are grouped into different topic groups, where the topic information is extracted from the transcripts of the video.

- *X. Zhu is with the Department of Computer Science, University of Vermont, 33 Colchester Ave., Votey 377, Burlington, VT 05401. E-mail: xqzhu@cs.uvm.edu.*
- *X. Wu is with the Department of Computer Science, University of Vermont, 33 Colchester Ave., Votey 351, Burlington, VT 05401. E-mail: xwu@cs.uvm.edu.*
- *A. Elmagarmid is with the Department of Computer Science, Purdue University, 250 N. University Street, West Lafayette, IN 47907. E-mail: ake@purdue.edu.*
- *Z. Feng and L. Wu are with the Department of Computer Science, Fudan University, 220 Handan Road, Shanghai 200433, P.R. China. E-mail: {zhfeng, ldwu}@fudan.edu.cn.*

Manuscript received 13 Oct. 2003; revised 15 Apr. 2004; accepted 20 Oct. 2004; published online 17 Mar. 2005.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0201-1003.

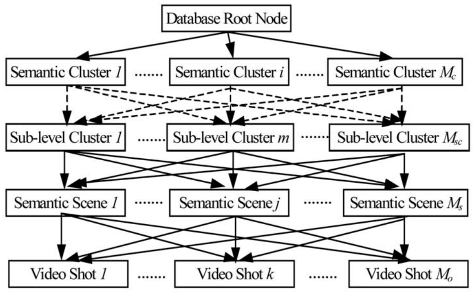


Fig. 1. The proposed hierarchical video database model.

3. Video association mining, where associations from video units are used to explore video knowledge [13], [14].

An intuitive solution for video mining is to apply existing data mining techniques [20], [21], [22] to video data directly. Nevertheless, as we can see from the three types of video mining techniques above, except [13], [14] which have integrated traditional sequential association mining techniques, most others provided their own mining algorithms. The reason is that almost all existing data mining approaches deal with various databases (like transaction data sets) in which the relationship between data items is explicitly given. Video and image databases (or other multimedia data) are different from them. The greatest distinction between video and image databases is that the relationship between any two of their items cannot be explicitly (or precisely) figured out. Although we may now retrieve video frames (and even physical shots) with satisfactory results, acquiring relationships among video frames (or shots) is still an open problem. This inherent complexity has suggested that mining knowledge from multimedia materials is even harder than from general databases [7], [23], [24], [44].

In this paper, we first introduce a knowledge-based video indexing framework to facilitate video database management and access. To explore video knowledge in supporting this framework, we propose a solution for a new research topic, video association mining, in which video processing and existing data mining algorithms are seamlessly integrated to mine video knowledge. We will systematically address the definitions and evaluation measures (temporal distance, temporal support, and confidence) for video associations by taking the distinct features of video data into consideration, and then proposing a solution in mining sequential patterns from the video stream that usually consists of multiple information sources (e.g., image, audio, and caption text). We use basketball videos as our test bed because sports video generates large interest and high impact worldwide.

The paper is organized as follows: In Section 2, we present a knowledge-based video indexing framework and introduce the system architecture for video association mining. We provide several techniques in Section 3 to explore visual and audio cues that can help us bridge the semantic gap between low-level features and video content. In Section 4, we present a video association mining scheme. We discuss algorithms to classify video associations and construct video indexing in Section 5. Section 6 presents the results of our performance evaluation.

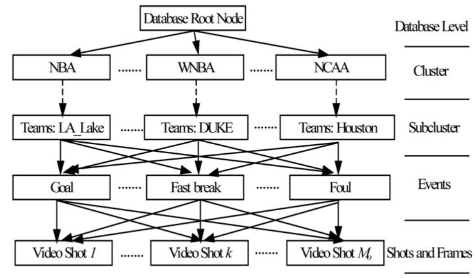


Fig. 2. Knowledge-based basketball video database management.

2 Knowledge-Based Video Indexing and System Architecture

There are two widely accepted approaches for accessing video in databases: shot-based and object-based. In this paper, we focus on the shot-based approach. In comparison with traditional video database systems that use low-level similarities among shots to construct indices, a semantic video database management framework has been proposed in Fig. 1, where video semantic units (scenes or story units) are used to construct database indices [7]. However, this scheme works on videos with content structure, e.g., movies and news, where video scenes are used to convey scenarios and content evolution. For many other videos, such as sports videos, there are no such story units. Instead, they contain various interesting events, e.g., a goal or a fast break, which could be taken as highlights and important semantics. Accordingly, by integrating the existing framework in Fig. 1, we propose a knowledge-based video indexing framework for basketball videos, as shown in Fig. 2. To support efficient video indexing, we need to address the following three key problems before we can actually adopt the framework in Fig. 2: 1) How many levels should be included in the model? 2) Which kinds of decision rules should be used at each node? and 3) Do these nodes make sense to human beings?

We solve the first and third problems by deriving knowledge from domain experts (or from extensive observations) and from the video concept hierarchy. For basketball videos, we first classify them into a two-level hierarchy. The first level is the host association of the games, e.g., NBA, NCAA, and CBA, and the second level consists of teams of each association, such as LA_Lake and Houston, where each video can be explicitly classified into one node. Then, we integrate the structure of video content to construct lower-level indices. As we have stated above, extensive observations and existing research efforts suggest that there are many interesting events in sports videos that can be used as highlights [16], [25], [26], [29]. For basketball videos, the events that likely attract most viewers' interests are goals, fast breaks, and free throws, etc. We can therefore use these events as nodes at the third level of our indexing structure. At the lowest level, we use the video shots as index nodes, as shown in Fig. 2, where each shot may have more than one parent node because some shots contain several events.

To solve the second problem, we find that the decision rules for the first two levels (cluster and subcluster) and the lowest level (shots and frames) are relatively easy and we can employ domain knowledge and some video shot segmentation algorithms [1], [27] to get satisfactory results. Our analysis in Section 3.2 also indicates that, by using the caption text in basketball videos, we can recognize team

(a) (b) (c) (d)

Fig. 3. Examples of the free throws of "foul shots," where shot (b) is captured right after shot (a).

names and their scores. Hence, the decision rules for the second level can also be accomplished automatically. Nevertheless, the most challenging task comes from the decision rules of the third level (events), i.e., mapping physical shots to various event nodes. In this paper, we will adopt video association mining to detect sports events. Our system architecture is given in Fig. 4, where various features are outlined below:

1. **A video association mining algorithm** to discover video knowledge. It also explores a new research area in video mining, where existing video processing techniques and data mining algorithms are seamlessly integrated to explore video content.
2. **An association-based video event detection scheme** to detect various sports events for database indexing. In comparison with other video event detection techniques, e.g., special pattern detection [25], the Hidden Markov Models [16], [29], and classification rules [28], the association-based technique does not need to define event models in advance. Instead, the association mining will help us explore models (associated patterns) from video.
3. **A knowledge-based sports video management framework** to support effective video access. The inherent hierarchical video classification and indexing structure can support a wide range of granularity levels. The organization of visual summaries is also inherently supported. Hence, a naive user can browse only a portion of highlights (events) to get a concise summary.

By integrating the video knowledge in the indexing structure, the constructed video database system will make more sense in supporting the retrieval and browsing for naive users. As shown in Fig. 3, where we provide four examples of "foul shots," it can be seen that the visual perception of these four shots vary a lot (especially for Fig. 3a and all others), but Fig. 3a and Fig. 3b both cover the same event of the same player, which are captured from different angles. With traditional video indexing mechanisms, these four shots will be indexed at different nodes (because they have different visual perceptions) and providing Fig. 3a as a query example may never work out results, like Fig. 3b (even if they do match with each other in semantics). With knowledge-based indexing, we can index them as one node (as long as we can detect this type of event), so the retrieval, browsing, and database management can be facilitated. When searching from a database constructed with the proposed indexing structure, the search engine can either include or exclude any index level to facilitate different types of queries. For example, if a user wants to query for a foul shot, regardless of the team names or the host association of the games (NBA, NCAA, etc.), the search engine can inherently attain this goal by ignoring the

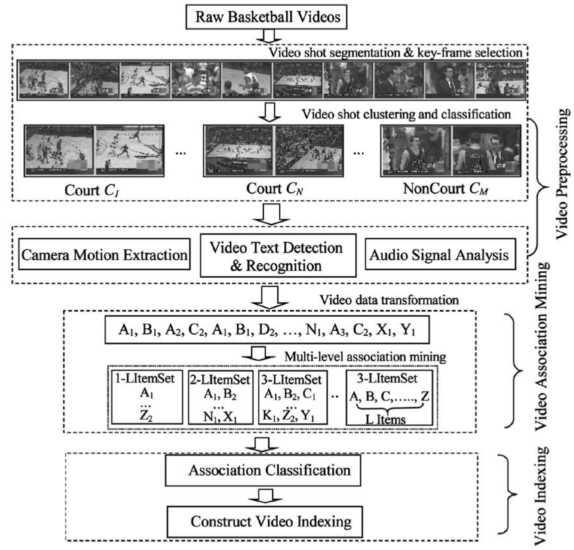


Fig. 4. The architecture of association-based video indexing.

first two levels of indexing (cluster and subcluster in Fig. 2) at the search stage.

In the system architecture in Fig. 4, we first parse a video sequence into physical shots and use a clustering algorithm to merge visually similar shots into groups. We then use dominant color detection to identify video groups that consist of court field shots and classify video shots into two categories: court and noncourt. We also perform camera motion extraction, audio signal analysis, and video text detection and recognition to detect visual and audio cues. A hybrid sequence is constructed by integrating the temporal order and the audio and visual cues of each shot. An association mining scheme is designed to mine sequential associations from the sequence. Finally, we classify all mined associations and use them to construct video indexing.

3 Video Preprocessing

To apply existing data mining techniques on video data, one of the most important steps is to transform video from nonrelational data into a relational data set. To facilitate this goal, we adopt a series of algorithms to explore audio and visual cues. We start with a raw video sequence and output symbolic sequences that indicate where and what types of cues appear in the video.

3.1 Video Shot Detection and Classification

Physical video shots that are implicitly related to content changes among frames are widely used in various video database systems [1]. To support shot-based video content access, we have developed a shot cut detection technique [27], which uses color features in each frame to characterize content changes among frames. The boundaries of shots are then determined by a threshold that is adjusted adaptively by using a small window (30 frames in our current work).

After shot segmentation, we try to classify each shot into two categories: court and noncourt. We first cluster visually similar shots into groups and then use the dominant color to identify groups which consist of court field shots because the court field in most sports can be described by one distinct dominant color [29]. To facilitate this goal, we use the 10th frame of each shot as its representative frame (key-frame)[1] and then extract two visual features from each key-frame (a 3D HSV color histogram and a 10-dimensional tamura coarseness texture [31]). When constructing a color histogram, we quantize H, S, and V into 16, 4, and 4 bins, respectively, so that the histogram of each image is characterized by a 256-dimensional vector and the total number of feature dimensions is 266. Given a video in the database, we assume it contains N shots $S_1, S_2, \ldots, S_N$ and denote the key-frame of S_i by K_i. Suppose $H_{i,l}$, $l \in [0, 255]$, and $TC_{i,n}$, $n \in [0, 9]$ are the normalized color histogram and texture of K_i. The distance between shots S_i and S_j is defined by (1), where W_C and W_T indicate the weight of each feature:

$$\begin{aligned} Dis(S_i, S_j) = W_c \left\{ 1 - \sum_{l=0}^{255} \min(H_{i,l}, H_{j,l}) \right\} \\ + W_T \sqrt{\sum_{n=0}^{9} (TC_{i,n} - TC_{j,n})^2}. \end{aligned} \tag{1}$$

We want to group shots that are similar into a cluster. In addition, different clusters should have sufficiently different characteristics. Hence, we adopt a modified *split-and-merge* clustering algorithm [32] by sequentially executing two major procedures: *merging* and *splitting*. In the *merging* procedure, we iteratively merge the most similar clusters (defined by (2)) until the distance between the most similar clusters is larger than a given threshold. Nevertheless, this *merging* procedure may generate clusters with a large intracluster distance (defined by (3)). Accordingly, after the *merging* procedure, we turn to the *splitting* procedure to split clusters with large visual variances. We iteratively calculate the intracluster distance for any cluster C_i, the cluster with its intracluster distance larger than a given threshold is separated into two clusters until all clusters have their intracluster distance less than the given threshold.

Let's denote the ith cluster by C_i and the number of members in C_i by N_i, where each element $(S_i^l, l = 1, \ldots, N_i)$ in the cluster is a shot. The intercluster distance between C_i and C_j is defined by (2):

$$d_{\min}(C_i, C_j) = \min_{S_i^l \in C_i, S_j^k \in C_j; l=1,\ldots,N_i, k=1,\ldots,N_j} Dis(S_i^l, S_j^k). \tag{2}$$

We then define the intracluster distance of C_i by (3):

$$d(C_i) = \max_{S_i^l \in C_i, S_i^k \in C_i;\ l \neq k;\ l=1,\ldots,N_i, k=1,\ldots,N_i} Dis(S_i^l, S_i^k). \tag{3}$$

After we have clustered visually distinct shots into groups, we can use the dominant color (usually, a tone of yellow) to identify groups that consist of court field shots. However, even though the color of the court field is likely a tone of yellow, the actual color may vary from stadium to stadium and also change with lighting conditions. Therefore, we cannot assume any specific value for this dominant color, but learn it adaptively. We randomly sample N frames from video sequences (in our system, we set $N = 50$). Because sports videos usually focus on the court field, most of these N frames will contain the court field. We then calculate the histogram of the hue component of each frame (in HSV color space). The histogram of the hue component is added up over these N frames. We pick up the peak of this cumulated hue histogram and use the corresponding hue value as the court field hue color. Assuming this hue color is denoted by $\bar{H}$, we calculate the average saturation and intensity value of the pixels in these N frames, where the hue color of the pixels is $\bar{H}$. We denote the average saturation and intensity by $\bar{S}$ and $\bar{I}$. For each group G_i (acquired from the former clustering algorithm), we calculate the dominant hue color of all key-frames in G_i, denote it by $\bar{H}_i$, and the average saturation and intensity of the pixels with their hue color equal to $\bar{H}_i$ are denoted by $\bar{S}_i$ and $\bar{I}_i$. Then, we use (4) to calculate the distance between G_i and the template. After we get the distances from all video groups, we use a simple thresholding method to classify each group into two exclusive categories: a group consisting of court filed shots or not:

1. For the sake of simplicity, we use this simplest key-frame selection mechanism. One can also adopt other complicated approaches [30]. Nevertheless, because our purpose is not to characterize the content change in the video shots, but to classify video shots into different categories, we find the performance of this simple mechanism works reasonably well.

Fig. 5. Video shot clustering results where each icon image represents one shot (the first row represents the first shot of each group and all other rows represent each clustered group)

$$HsvDis(i) = \sqrt{(\bar{I}_i - \bar{I})^2 + (\bar{S}_i)^2 + (\bar{S})^2 - 2\bar{S}_i \cdot \bar{S} \cdot \cos(\theta)}, \tag{4}$$

$$\theta = \begin{cases} |\bar{H}_i - \bar{H}| & if\ |\bar{H}_i - \bar{H}| < 180^\circ \\ 360^\circ - |\bar{H}_i - \bar{H}| & if\ |\bar{H}_i - \bar{H}| > 180^\circ. \end{cases} \tag{5}$$

Generally, since one sports video is captured from one place, both shot clustering and classification can acquire relatively good performances. As shown in Fig. 5, we can find that the shots containing the court are successfully clustered into groups (and likely characterized by cameras with different angles or views) because the court field color plays an important role in similarity evaluation.

3.2 Video Text Detection and Recognition

There are two types of video text: the first is the text shown in video scenes, referred to as scene text hereafter, and the second is the text postprocessed and added into the video, such as team names and their scores, which we call caption text. For sports videos, caption text is much more important than scene text because the former directly conveys video semantics. With caption text, we can acquire the name of each team and use it to construct the second level index in Fig. 2. Moreover, as long as we can detect the team scores, the score change is directly associated to the "goal" events.

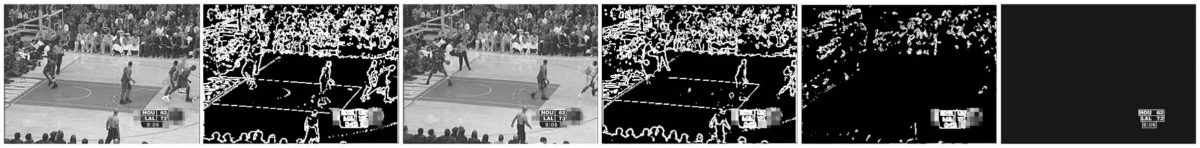


Fig. 6. Video caption text region detection, (a) frame F_i, (b) edge frame E_i, (c) frame $F_{i+\tau}$, (d), edge frame $E_{i+\tau}$, (e) the edge pixels which appear in both E_i and $E_{i+\tau}$, and (f) detected caption text regions.

In comparison with scene text, the caption text has one distinct feature: It rarely moves. This distinct feature inspires us to develop a simple but efficient caption text detection algorithm:

1. Calculate the edge of the current frame F_i, denote the edge frame by E_i, and then calculate the edge of the succeeding frame with a step τ (in our system, we set $\tau = 10$), i.e, frame $F_{i+\tau}$ and its edge frame $E_{i+\tau}$.
2. Compare edge pixels in E_i and $E_{i+\tau}$. If the edge pixel in E_i is still the edge pixel in $E_{i+\tau}$, the current pixel is a candidate of caption text pixel.
3. After all edge pixels in E_i have been processed, use a median filter to eliminate noise and all remaining pixels to form the caption text regions.

If the camera motion were still, we take the locations of the text regions detected from the most recent moving frame as the caption text regions in the current frame because, without moving the camera, all edge pixels in E_i and $E_{i+\tau}$ are the same and the proposed method may not work. Meanwhile, we add another constraint: The detected caption text region should appear in either the top $\frac{1}{4}$ or bottom $\frac{1}{4}$ of the frame region. We have observed various basketball videos from ESPN, FOX, etc., and found that, in almost all situations, the team names and their scores appear in the top or the bottom regions of the frame because it has less impact on the viewers.

After candidate text regions have been detected, we need to prune some false candidates and handle the scale problem. The regions with their height and width less than given thresholds are eliminated and the horizontal-vertical ratio of the regions should also be in a certain range. After that, we use the Bilinear Interpolation algorithm to resize each candidate region into a certain size of box and transform the pixels into binary values (black or white) for recognition.

To recognize caption text, we adopt an existing OCR (Optical Character Recognition) engine, WOCAR [33], which takes a binarized image as the input and yields an ASCII string result. This engine has many function calls to support applications. More details about video text detection can be found in [34]. Fig. 6 gives an example of our caption text detection results. Meanwhile, since we only detect team names and score numbers, we can develop a small vocabulary for the OCR engine to improve the recognition accuracy. We perform the algorithm on every τ ($\tau = 10$) frames and use detected team names to construct the second level index. Once we detect a score change, we add a symbolic tag at the corresponding place.

3.3 Camera Motion Characterization

Given a shot S_i, the camera motions in the shot can also imply some knowledge. For example, a fast break usually happens when the camera is still, or pans slowly, then suddenly speeds up and pans quickly. Hence, we can explore semantic cues from camera motions in each shot. However, the camera motions in noncourt field shots have less knowledge or can even be meaningless. We therefore only analyze camera motions from court field shots.

To extract camera motions, we have recently developed a qualitative camera motion extraction method [35]. This method works on compressed MPEG streams and uses motion vectors from P-frames to characterize camera motions. For any two motion vectors in each P-frame, we first classify their mutual relationship into four categories: approaching, parallel, diverging, and rotation, as shown in Fig. 7. Generally, if the camera pans or tilts, the mutual relationship between any two motion vectors is likely parallel, as shown in Fig. 9 and, if the camera zooms, the mutual relationship is likely to be approaching or diverging (depending on whether the actual motion is zoom-in or zoom-out). We then construct a 14-bin motion feature vector to characterize the camera motion in each P-frame. More details related to the camera motion classification can be found in [35]. Only certain types of camera motions in basketball videos could possibly imply useful information and we therefore classify the camera motion of each P-frame into the following six categories: Still, Pan (left and right), Zoom (in and out), and others. A motion description hierarchy is given in Fig. 8.

In addition to classifying the camera motion, we also calculate the average motion magnitude of each P-frame by (6), where MV_i is the number of valid motion vectors in the P-frame i, $x_i(m)$ and $y_i(m)$ are the x and y components of the motion vector m in the frame i. Our objective is to characterize the speed of motion activities. We roughly classify the motion magnitude into three categories: slow, medium, and fast, by specifying a numeric range for each category. Finally, a temporal filter is adopted to eliminate falsely detected camera motions. For the MPEG videos used in our test bed, there are eight P-frames in each second of stream. So, we use the dominant motion of these eight P-frames and its magnitude as the camera motion in this range and collect camera motions and magnitude (in the original temporal order) to form a symbolic sequence. For

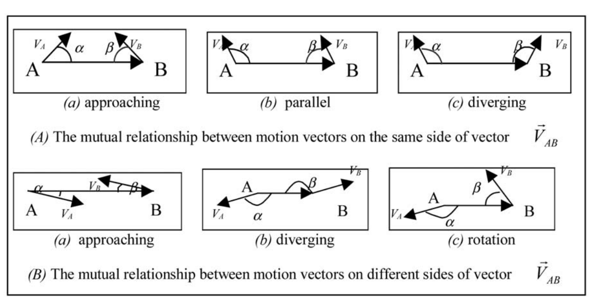


Fig. 7. Mutual relationships between two motion vector in each P-frame.

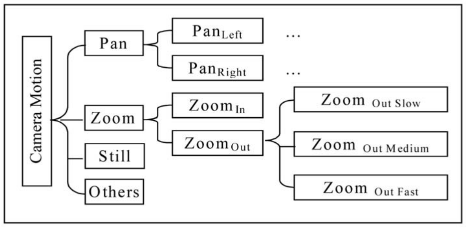


Fig. 8. Camera motion description hierarchy.

(a) (b) (c)

Fig. 9. Camera pan operation between two frames (a) and (b), and (c) the corresponding motion vectors.

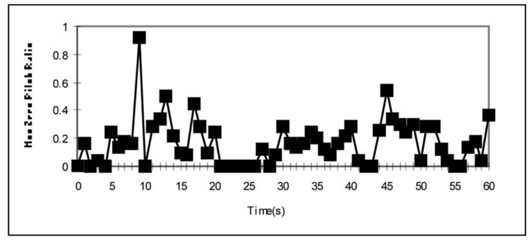


Fig. 10. Nonzero pitch ratio from an audio signal.

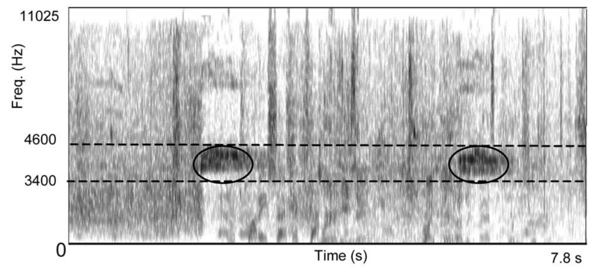


Fig. 11. Spectrum of an audio signal with whistle.

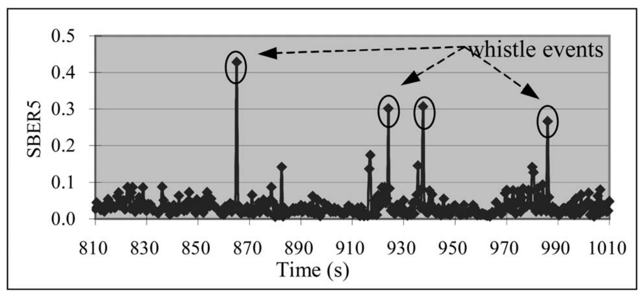


Fig. 12. Segment band energy ratio of the fifth subband from an audio with multiple whistle events.

MPEG videos encoded with fewer P-frames, one can use a longer time span for temporal filtering, because the dominant camera motion in sports video usually lasts several seconds. With the proposed approach, we can identify three typical camera motions: Pan, Tilt, and Zoom. All other camera motions are marked as "Others." For mining purposes, after camera motion detection all "Others" tags will be removed from the sequence:

$$M(i) = \sum_{m=1}^{MV_i} \sqrt{x_i(m)^2 + y_i(m)^2} \Big/ MV_i. \tag{6}$$

3.4 Salient Audio Event Detection

In sports videos, some special audio events, e.g., audience applause and a referee's whistle, will help us acquire some semantic cues. Generally, audience applause occurs when exciting events happen, e.g., shooting and/or a goal, and a referee's whistle may imply an interruption or another special event.

To detect audience cheering, we use the pitch of audio signal. Basically, pitch is the fundamental frequency that reveals harmonic properties of audio and is an important parameter in the analysis and synthesis of speech signals. In comparison with voice and music, the pitch value of audience applause is very small. In most cases, this value in sports videos is zero because, when cheering happens, the audio signal exhibits a constant high value noise that likely drowns out other audio signals, e.g., the voice of the anchorperson or the music. We therefore extract the pitch for each audio frame. In our system, the audio frame length is 20ms and the frame shift is 0ms. Because the duration of cheering usually exceeds 1 second, we apply cheering detection on each 1-second segment. For each segment, we calculate the NonZero Pitch Ratio (*NZPR*), which is defined as the ratio between the number of frames whose pitch is not zero and the total number of frames in a segment. For a cheering segment, its *NZPR* value likely exhibits a small value, and a simple threshold scheme can distinguish cheering segments from others. Fig. 10 shows the results of *NZPR* values from a test sports video with one minute duration, where four cheering events appear at 3s-9s, 20s-25s, 41s-44s, and 54s-57s.

To detect a referee's whistle, we use spectrum domain features. Fig. 11 demonstrates the spectrum of an audio segment that contains two whistles. The regions with a circle margin correspond to the spectrum when the referee whistles. One can find that, in frequency regions between 3500Hz to 4500Hz, the energy of a whistle is much higher than others. We then calculate the energy ratio between frequency 3500Hz and 4500Hz for each audio frame to detect whistles. We split the whole frequency into B subbands. Given audio frame i and subband j, we define the band energy ratio (*BER*) by (7), where $DFT_{i,k}$ is the *Discrete Fourier Transformation* of the audio frame i and E is the order of *DFT* coefficients. In our system, the sampling rate for audio signals is 22050Hz and B is 12. Thus, the frequency of the fifth subband is 3675 $\sim$ 4594Hz. Then, we calculate the segment band energy ratio of the fifth subband $(SBER_5)$ during a short time period (0.5s) by (8), where AF is the total number of audio frames in this period. Fig. 12 shows the results of $SBER_5$ values from a test sports video of about 200 seconds in length. The regions with a circle margin correspond to whistle events. We can then involve some thresholding mechanisms to find out the location of those whistle events.

$$BER_{i,j} = \sum\nolimits_{e=\frac{E}{B}(j-1)}^{\frac{E}{B}j} DFT_{i,e} \Big/ \sum\nolimits_{e=1}^{E} DFT_{i,e}, \tag{7}$$

$$SBER_5 = \frac{1}{AF} \sum_{i=1}^{AF} BER_{i,5}. \tag{8}$$

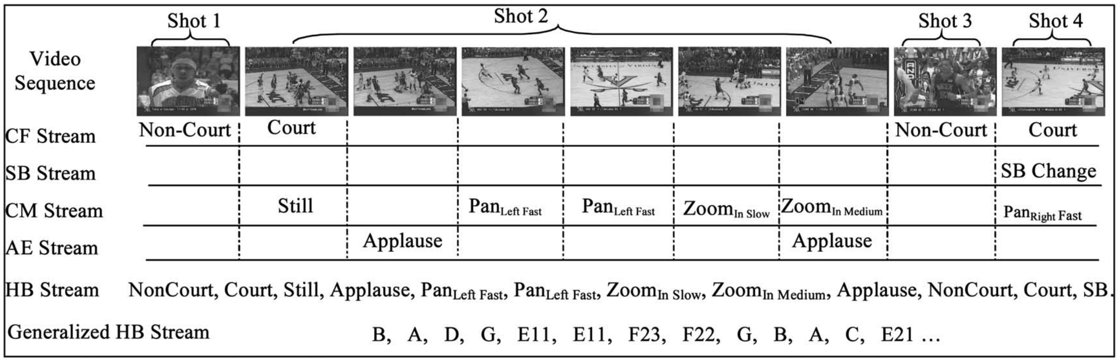


Fig. 13. Video data transformation and generalization.

TABLE 1
A Mapping Table to Generalize Video Data

Streams	CF Stream		SB Stream	CM Stream					AE Stream	
	Court	Non-court	SB Change	Still	$Pan_{Left\ Fast}$	$Pan_{Left\ Med.}$	…	$Zoom_{In\ Slow}$	Applause	Whistle
Generalization	A	B	C	D	E11	E12	…	F23	G	H

4 Association Mining from Video Data

Generally, there are two types of videos in our daily life: videos with some content structure and videos without any content structure. The former are videos such as movies and news where scenarios are used to convey video content. In [13], [14], we have proposed techniques to mine associations from this type of videos. For videos without content structure, e.g., sports videos, associations may still exist where the associations could be characterized as a series of sequentially related actions. For example, in basketball videos, a series of actions, such as Camera pan $\rightarrow$ Camera still $\rightarrow$ Camera zoom-in $\rightarrow$ Applause $\rightarrow$ Scoreboard change, likely appear sequentially, because they usually accompany a goal event. Mining associations from these videos, which do not have content structure, will not only facilitate knowledge acquisition, but also help us in realizing intelligent video management. In this section, we discuss techniques for video association mining, where the definitions and measures for video associations, and the sequential pattern search procedure are extensively studied.

4.1 Video Data Transformation

With techniques in Section 3, the original video sequence is transformed into four separated symbolic streams: court field (CF), camera motion (CM), scoreboard (SB), and audio events (AE), as shown in Fig. 13. Our next step is to conduct mining activities on these streams. To this end, there are two solutions: Treat data streams separately or combine them together as a single stream. Oates and Cohen [36] proposed a mechanism which treats multiple streams separately when conducting the mining activity, where the objective is to find the cooccurrence of the patterns that appear in the multiple streams. However, this method requires that the streams which take part in the mining activity be synchronized, where each stream produces the same amount of symbols in the same amount of time. In our situation, the multiple streams extracted from video data obviously do not satisfy this requirement. Intuitively, combining multiple streams into a single stream appears to be an easier way for data mining purposes because mining from one stream is obviously easier than mining from multiple sources and many research efforts have been conducted to find patterns, e.g., periodic patterns [37], [45], from a data stream. However, we need to guarantee that there is no information loss when combining multiple streams, which means that, after the data combination, we should maintain the original temporal order information of each separate stream in the combined stream. To this end, we adopt the following approach to combine multiple symbolic steams into a single hybrid (HB) stream: 1) For video and audio cues which happen at different time slots, we put all their tags together, with each tag placed at a corresponding place in its original stream. 2) If multiple tags happen at the same time, e.g., a scoreboard change and a camera motion happen at the same time, we use the same order to combine them in all situations, e.g., a scoreboard change always precedes a camera motion. An example of video data transformation is shown in Fig. 13, where information from four separate streams is combined to form a hybrid stream.[2] With such a mechanism, the temporal order information in each separate stream is well maintained in the transferred hybrid stream and combining multiple streams into a single stream will not lose information for effective association mining from data streams.

We have adopted a hierarchical camera motion description in Fig. 8, so we have to generalize an *HB* stream for multilevel association mining. Our generalization is accomplished by assigning a symbol to each type of tag, as shown in Table 1. For events with a hierarchy, we generalize them into a set of characters with each character indicating a state. For example, for "E12," "E" denotes camera pan, "1"

2. We mark only one CF tag for each video shot, which is placed at the beginning of the shot, because a shot either belongs to the court field or not. Inside each shot, we will analyze its content and explore other video and audio cues. This is the reason that some video shots receive several tags, as shown in shot 2 of Fig. 13. This is different from the statements in Section 3.1, where only one key-frame is extracted from each shot to classify a video shot into a noncourt shot or a court shot.

indicates the panning direction, and "2" represents the motion magnitude. In Fig. 13, the last row gives a generalized *HB* stream.

4.2 Definitions and Terminology

Based on the above observations, we define a video association as a sequential pattern with $\{X_1..X_i..X_L;\ X_i^t < X_j^t$ for any $i < j\}$, where X_i is a video item (see Definition 1 below), L denotes the length of the association, $X_1 \cap .. \cap X_i .. \cap X_L = \oslash$, X_i^t denotes the temporal order of X_i, and $X_i^t < X_j^t$ indicates that X_i happens before X_j. For simplicity, we use $\{X\}$ as the abbreviation for a video association.

Generally, two measures (*support* and *confidence*) have been used to evaluate the quality of an association. However, these measures do not consider temporal information of the items in the association. For video associations, the *temporal distance* (see Definition 5 below) between neighboring items implies some useful information: The smaller the temporal distance between neighboring items, the larger is their correlation. For example, if two neighboring shots contain applause and scoreboard change, respectively, we naturally believe that they are correlated. However, the applause that happens several shots (e.g., three more shots) before the scoreboard change rarely indicates any correlation between them. That is, for associations with a large temporal distance between neighboring items, their items usually have a weaker correlation and, therefore, can imply almost no knowledge. Accordingly, instead of using the traditional support measure, we adopt a *temporal support* (*TS*) to evaluate the video association. Moreover, several other definitions are also given below:

1. A video *item* is a basic unit in association mining. In this paper, it denotes a symbolic tag acquired from video processing techniques, i.e., a symbolic unit in the hybrid video stream.
2. An *L-ItemAssociation* is an association that consists of L sequential items. For example, "AB" is a 2-Item-Association and "ABC" is a 3-ItemAssociation.
3. An ItemSet is an aggregation which consists of video associations. More specifically, an *L-ItemSet* is an aggregation of all *L-ItemAssociations*, each of which is an *L-ItemAssociation*.
4. *L-LItemSet* is an aggregation of all *L-ItemAssociations* whose *temporal support* (see Definition 7 below) is no less than a given threshold.
5. Given a transformed hybrid video stream, the *temporal distance* (*TD*) between two items is the temporal identification difference of the shots that contain these two items. For example, in the hybrid stream demonstrated in Fig. 14, the first time the pattern $\{AB\}$ appears, their temporal distance $TD\{AB\}$ is 0 because they happen in the same shot. The second time $\{AB\}$ appears, $TD\{AB\}$ equals 1 because A and B happen in two neighboring shots and the temporal identification difference between the neighboring shots is 1.
6. The *temporal distance threshold* (*TDT*) specifies the upper bound that the *temporal distance* must comply with, i.e., no larger than this threshold. Take the pattern $\{AB\}$ in Fig. 14, for example, when $TDT = 1$, $TD\{AB\} = 2$ will not satisfy because $TD\{AB\} = 2$ is larger than the given *TDT* value.

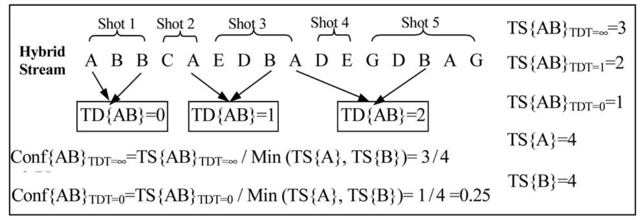


Fig. 14. Example of video association evaluation in terms of temporal support and confidence

7. Given a *temporal distance threshold* (*TDT*) $TDT = T$, the *temporal support* (TS) of an association $\{X_1 \ldots X_L\}$ is defined as the number of times this association appears sequentially in the sequence. In addition, each time this association appears, the temporal distance between any two neighboring items of the association should satisfy the given TDT (i.e., no more than T shots). In Fig. 14, when $TDT = \infty$ (i.e., ignoring the temporal distance), the temporal support for $\{AB\}$ is $TS\{AB\} = 3$. However, when we set $TDT = 1$, $TS\{AB\}$ becomes 2 because the last time $\{AB\}$ appears, its temporal distance $(TD\{AB\} = 2)$ does not satisfy the given TDT. It is obvious that the smaller the TDT, the stronger the semantic correlations among the mined associations are.
8. Given $TDT = T$, the *confidence* of an association $\{X_1, ., X_L\}$ is defined as the ratio between the temporal support of $\{X\}$ (when $TDT = T$) and the maximal number of possible occurrences of the association $\{X\}$. Because the maximal possible occurrences of the association are determined by the number of occurrences of the item with the minimal support, the confidence of the association is defined by (9). The examples of the confidence evaluation have been provided in Fig. 14, where different TDT values result in different confidences for the same association. The larger the confidence value, the more confidently the association holds:

$$Conf\{X\}_{TDT=T} = TS\{X\}_{TDT=T} / Min(TS(X_1), .., TS(X_L)). \tag{9}$$

4.3 Video Association Mining

4.3.1 Multilevel Associations

We have introduced a hierarchy in Fig. 8 (which can also be interpreted as a taxonomy) to characterize camera motions. When a taxonomy exists, the supports of associations at lower levels are lower than associations at higher levels. Accordingly, some solutions have been proposed to mine multilevel associations [38]. The motivation behind these algorithms is simple and intuitive: For all hierarchical items, their ancestors at higher levels are added into data sets and a data mining algorithm is executed on new data sets for multiple times to mine multilevel associations. As shown in Table 2, given a generalized *HB* sequence in Tables 2a, 2b, and 2c, show the 1-ItemSet at level 1 and level 2, respectively. As we can see, only the descendants of the large ItemSet at level 1 are considered as candidates for level 2 large 1-ItemSet.

TABLE 2
Multilevel Association Mining: (a) A Generalized *HB* Sequence, (b) 1-ItemSet at Level 1, and (c) 1-ItemSet at Level 2

Generalized HB sequence
F11, A, E13, C, F22, F23 E12, D, F21, A, E23, C, A, F13

(a)

1-ItemSet	Suppor
{A}	3
{E**}	3
{F**}	5

(b)

1-ItemSet	Support
{A}	3
{E1*}	2
{F1*}	2

(c)

4.3.2 The Mining Algorithm

Our video association mining algorithm consists of the following phases:

1. **Transform**. This phase adopts various techniques to explore visual and audio cues and transforms video data into a relational data set D.
2. **L-LItemSet**. In this phase, we mine video associations with various levels and lengths. We first find an L-ItemSet and then use the L-ItemSet and user-specified thresholds to find L-LItemSet. We iteratively execute this phase until no more nonempty L-LItemSet can be found.
3. **Collection and Postprocessing**. This phase collects and postprocesses video associations for different applications.

We have discussed techniques for Phase 1 and Phase 3 directly relates to applications of video associations, which is trivial from the data mining point of view. Therefore, we focus on Phase 2 only, where its main procedure is shown in Fig. 15. Throughout this section, we use the notions that l denotes the level of associations (the maximal level of associations *max_level* is 3 in our system) and $D[l]$ represents the filtered data set at level l. $I[l,k]$ and $L[l,k]$ are the aggregations of k-ItemSet and k-LItemSet at level l, respectively. $\{\mathrm{X}\}.\mathrm{Item}_k$ means the kth item of the association $\{X\}$.

Basically, Phase 2 consists of two stages: 1) In the first stage, the algorithm filters the data set at level l and uses the filtered data set $D[l]$ to construct 1-ItemSet and 1-LItemSet at level l, as shown on lines 2 to 4 in Fig. 15. 2) Then, the algorithm uses the constructed 1-LItemSet and the candidate generation procedure (Fig. 16) to progressively mine k-ItemAssociations, $k = 2, 3, \ldots,$ at level l, until the constructed k-LItemSet at level l is empty. Then, the algorithm turns to next level $l + 1$ and mines associations at this level.

As shown in Fig. 15, for each level l, we first filter the data set D, $Filter\_Dataset(D, l)$, to process items that are no larger than level l. For example, when $l = 2$, this procedure filters items $\{E13, E12\}$ as $\{E1, E1\}$ and the higher the level, the more subtle the filtered item is. The filtered sequence is put into a new data set $D[l]$. We then use $D[l]$ to generate 1-ItemAssociations at level l (denoted by $I[l, 1]$) by using function $Get\_1\_ItemSet(D[l], l)$. We use the generated 1-ItemSet and the user specified minimal support $minSup[l]$ to generate 1-LItemSet at level l (denoted by $L[l, 1]$) with procedure $Get\_1\_LItemSet(D[l], I[l, 1], minSup[l])$. The generated 1-LItemSet consists of associations in 1-ItemSet which satisfy the user-specified minimal support $minSup[l]$. Because 1-ItemAssocitions do not involve any temporal distance, we ignore TDT when constructing the 1-LItemSet. We then use the generated 1-LItemSet at level l to mine associations with larger lengths. This is facilitated by adopting an *Apriori*-like algorithm which uses multiple passes to generate candidates and evaluate their supports.

In each pass, we use the LItemSet from the previous pass to generate the candidate ItemSet and then measure the temporal support of generated candidates by making a pass over the database $D[l]$. At the end of the pass, the support of each candidate is used to determine the frequent ItemSet.

Candidate generation for each pass is similar to the method in [12]. It takes the set of all $k-1$-$ItemAssociations$ in $L[l, k-1]$ and all their items as input and works as shown in Fig. 16. The items in $L[l, k-1]$ first join together to form new candidates. To this end, for any two distinct $k-$ 1-ItemAssociations $\{p\}$ and $\{q\}$ in $L[l, k-1]$, if their first $k-2$ items are the same (as shown on line 3 in Fig. 16), we will generate a new k-ItemAssociation $\{X\}$. The first $k-2$ items of $\{X\}$ are the same as that of $\{p\}$ and the $k-1\mathrm{th}$ and kth items of $\{X\}$ are the $k-1\mathrm{th}$ item of $\{p\}$ and $\{q\}$, respectively (as shown on line 5 in Fig. 16). Then, $\{X\}$ is taken as a candidate and put in $I[l, k]$. We iteratively repeat the same procedure until all elements in $L[l, k-1]$ have been evaluated. After that, we prune out the candidates in $I[l, k]$ whose subsequences are not in $L[l, k-1]$ because, if a

```
Procedure VAMining ()
Input: (1) Hybrid data steam D; (2) max. association level max_level; and (3) TDT and minimal support and confidence in
        selecting associations at different levels minSup[l], minConf[l], l=1,…, max_level.
Output: Mined multi-level video associations
(1)   For (l=1; l ≤ max_level; l++)                               // mine associations at various levels
(2)      D[l] = Filter_Dataset (D, l);                            // process items that are no larger than level l
(3)      I [l, 1]= Get_1_ItemSet (D [l], l)                       // find 1-ItemSet at level l
(4)      L[l, 1]=Get_1_LItemSet (D[l], I[l,1], minSup[l])         // find 1-LItemSet at level l
(5)      For ( k = 2; L[l, k-1] ≠ Ø; k ++ )                       // mine associations with different lengths
(6)         I [l, k]=CandidateGeneration(L[l, k-1])               //generate candidates, see Fig. 16
(7)         L[l, k] ← Φ                                           // initialize k-LItemSet
(8)         For each k-ItemAssociation {X} in I [l, k], {X}∈ I [l, k]   //evaluate each generated candidate
(9)            TS{X}_TDT= Calculate_TS (D [l], TDT)               // calculate temporal support of X
(10)           Conf{X}_TDT= TS{X}_TDT / Min (TS{X_1},.., TS{X_k})  // calculate confidence of X
(11)           L[l, k]←L[l, k] ∪ {X}| {X}∈I [l, k],  TS{X}_TDT ≥ minSup[l] & Conf{X}_TDT ≥ minConf[l]
                        //select associations with their temporal support and confidence larger than the given thresholds
            Endfor
```

Fig. 15. Pseudocode for multilevel video association mining.

```
Procedure CandidateGeneration()
Input: L[l, k-1]     // k-1 large item set at level l   Output: I[l, k]        // k item set at level l
(1).   I[l, k] ← Φ
(2).   For any two k-1-ItemAssocations {p} and {q} in L[l, k-1]          //Join L[l, k-1] with L[l, k-1]
(3).   |  If ({p} ≠ {q}) and ({p}.Item1={q}.Item1 & ..& {p}.Itemk-2={q}.Itemk-2)
(4).   |  |  Generate a k-ItemAssociation {X} such that:
(5).   |  |     {X}.Item1={p}.Item1, &…&, {X}.Itemk-2={p}.Itemk-2, {X}.Itemk-1={p}.Itemk-1, {X}.Itemk={q}.Itemk-1
       |  |  I[l, k] ← I[l, k] ∪ {X}
(6).   |  End If
(7).   End For
(8).   Delete any member {X} ∈ I [l, k] such that some k-1-ItemAssociation of {X} is not in L[l, k-1].
```

Fig. 16. Pseudocode for candidate generation.

TABLE 3
An Example of Video Association Mining, where $\{X\}_S^C$ Indicates an Association

Hybrid Stream D (left to right, top to bottom)	2-LItemSet (TDT=1)	2-LItemSet (TDT=∞)	3-LItemSet (TDT=∞)	Candidate 4-ItemSet (after join)	4-ItemSet (after pruning)	4-LItemSet (TDT=∞)
A B C B C D C A C B C A B D A B C D B E	$\{AB\}_4^{1.0}, \{AC\}_3^{0.8}, \{AD\}_1^{0.3}, \{BA\}_2^{0.5}, \{BB\}_1^{0.3}, \{BC\}_4^{0.7}, \{BD\}_3^{1.0}, \{CB\}_4^{0.7}, \{CC\}_2^{0.7}, \{CD\}_2^{0.7}, \{DB\}_2^{0.7}$	$\{AB\}_4^{1.0}, \{AC\}_3^{0.8}, \{AD\}_3^{1.0}, \{BA\}_3^{0.8}, \{BB\}_3^{1.0}, \{BC\}_4^{0.7}, \{BD\}_3^{1.0}, \{CB\}_4^{0.7}, \{CC\}_3^{1.0}, \{CD\}_3^{1.0}, \{DB\}_3^{1.0}$	$\{ABC\}_3^{0.8}, \{ABD\}_3^{1.0}, \{ACB\}_3^{0.8}, \{ACD\}_3^{1.0}, \{BCD\}_3^{1.0}, \{CDB\}_3^{1.0}$	$\{ABCD\}$ $\{ABDC\}$ $\{ACBD\}$ $\{ACDB\}$	$\{ABCD\}$ $\{ACDB\}$	$\{ABCD\}_3^{1.0}$

X denotes the items of the association, S and C indicate the temporal support, and confidence of the association, respectively (for simplicity, we assume each video shot has only one symbolic tag and the HB stream has only one level).

subsequence of an association is not frequent, this association will not be frequent neither. All remaining candidates are taken as associations in $I[l, k]$. Table 3 provides an example of candidate generation, where the fourth column gives the 3-LItemSet and the fifth column is the join results (candidates) from the 3-LItemSet. After pruning out sequences whose subsequences are not in the 3-LItemSet, the sequences shown in the sixth column will be left. For example, {ABDC} is pruned out because its subsequence {BDC} is not in the 3-LItemSet.

4.3.3 Search Patterns from Hybrid Stream with Constraints

To mine video associations, the most important procedure is to search the appearances of the candidate pattern in the data stream, and this problem is complicated by users' constraint on the temporal distance (TDT) between items of the pattern. For example, with the HB stream in Fig. 17a, when searching the appearance for pattern $\{AEFBG\}$, many other approaches [37], [39] usually adopt a sliding window (e.g., w_1 and w_2 in Fig. 17a) to evaluate whether the pattern appears in the window or not. Such a windowing procedure has two obvious disadvantages: 1) Users have no control with the temporal distance between the items of the pattern, i.e., this approach ignores the temporal distances in the pattern, and 2) users have to well define the width of the window, otherwise the pattern may never fall into any window.

Accordingly, we need to design a new search mechanism by considering the *temporal distance* between neighboring items of the pattern. The simplest solution for this problem is to adopt a *waiting-and-matching* [46] method: We start from the first item of the pattern $\{AEFBG\}$ and scan the data stream until the certain item appears; at any state, if the *temporal distance* between items violates the TDT, the search procedure restarts. In Fig. 17a, "choice 1" of $\{AEFBG\}$ represents the results from this method. This approach, however, could miss targets if the user specifies a relatively small TDT. In Fig. 17a, if we set $TDT = 2$, the *waiting-and-matching* mechanism will fail to find the pattern because the *temporal support* between "BG" in "choice 1" is 3, which is larger than $TDT = 2$. However, there are other choices that $\{AEFBG\}$ actually satisfies $TDT = 2$, e.g., "choice 2" and "choice3."

Motivated by the above observations, we propose a new algorithm for searching sequential patterns from data stream with constraints. The intuitive idea behind this scheme is to push an item backward as much as we can

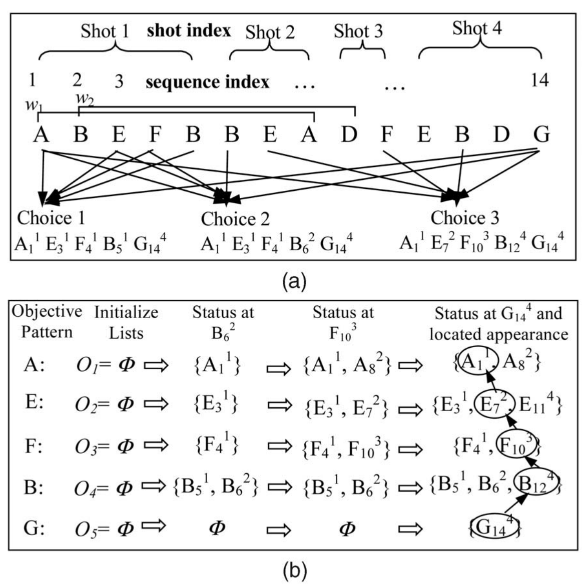


Fig. 17. Search candidates from a hybrid stream, where X_i^j represents the index information of the item X (j means in which shot the item appears and i indicates the order of the item in the stream). (a) An example of hybrid stream. (b) Search procedure (TDT=2).

(without violating the TDT), so we can maximize the possibility that, under the constraint of TDT, the pattern may appear in the stream. The algorithm consists of following major steps:

1. Given a pattern $\{X_1, X_2.., X_L\}$, a hybrid stream D, and a user-specified TDT, we call D $\{X_1, X_2.., X_L\}$ the objective pattern in. For each item X_i in the objective pattern, we construct a list O_i to record the appearances of X_i in D and initialize the list with $O_1 \leftarrow \Phi, .., O_i \leftarrow \Phi; , .., O_L \leftarrow \Phi; .$
2. Starting from the first item of the objective pattern, for each item $X_i, i = 1, ..L$, we search the first appearance of X_i from D (and ignore the appearance of any other item X_j, $j > i$). If item X_i appears in D, we put the index (sequence index and shot index) of the appearance into the list O_i. As demonstrated in Fig. 17b, $O_1 \leftarrow O_1 \cup A_1^1$; $O_2 \leftarrow O_2 \cup E_3^1$, and so on. This procedure continues until all items $X_i, i = 1, .., L$, have at least one member in their list $O_i, i = 1, .., L$.
3. When searching the appearance of the current item X_i, if any former item (including X_i itself) X_j, $j \leq i$ appears in D again, we put the index of X_j in the list O_j as long as the appearance of X_j satisfies the TDT. As shown at "status in B_6^2" in Fig. 17b, when searching for the appearance of "G," another "B" comes. Denote O_i^k by the kth member in the list O_i, and T_i by the number of members in O_i, so $O_i^{T_i}$ is the last member in O_i. To evaluate whether the appearance of X_j satisfies the constraint of TDT, we calculate two measures: a) the *temporal distance* between X_j and the latest appearance of its neighboring item $O_{j-1}^{T_{j-1}}$, $TD(X_j, O_{j-1}^{T_{j-1}})$ and b) the *temporal distance* between the current location of X_j and the last member in O_{i-1}, $TD(X_j, O_{i-1}^{T_{i-1}})$. If $TD(X_j, O_{j-1}^{T_{j-1}}) \leq TDT$, we add the index of X_j into its list O_j and continue the procedure; otherwise, O_j remains unchanged. Meanwhile, if $TD(X_j, O_{i-1}^{T_{i-1}}) > TDT$, it will indicate that, even if we assume the item X_i does appear at the current location of X_j, the temporal distance with its neighboring item X_{i-1} still violates the constraint of TDT, so there is no need to search the appearance of X_i any further. We will restart searching the appearance of the objective pattern from the location of the last member in O_1. Meanwhile, all lists should be initialized with $O_1 \leftarrow \Phi, .., O_i \leftarrow \Phi, ..O_L \leftarrow \Phi$.
4. As long as the lists of all items $(O_1, O_2, .., O_L)$ have at least one member, we cease the current search procedure because an appearance of the pattern have been located so far. As shown at "status at $G_{14}^4 \ldots$" in Fig. 17b, we will start from the last member in O_L (actually, there is only one member in O_L), denote it by O_L^*, and check all members in O_{L-1} in an inverse order (backward) to find the member that appears before O_L^* and has the smallest *temporal distance* with O_L^*. We denote this member by O_{L-1}^* and then find the member from O_{L-2} that appears before O_{L-1}^* and has the smallest temporal distance with O_{L-1}^*. We repeat the same procedure until the appearances of all items have been located. The sequence $\{O_1^*, .., O_L^*\}$ will provide actual locations of the pattern, as shown in Fig. 17b. Then, we initialize all lists with $O_1 \leftarrow \Phi, .., O_i \leftarrow \Phi, ..O_L \leftarrow \Phi$ and restart to locate the next appearance of the pattern from the location next to O_L^*.

As shown in Fig. 17a, no matter what TDT value (1, 2, or 3) users specify, our algorithm will exactly locate only one location for $\{AEFBG\}$, which is "choice 3." However, with the *waiting-and-matching* approach, only "choice 1" could be found and, if we set $TDT = 1$, it will miss the appearance of the pattern because, in this case, "choice 1" does not satisfy the TDT. Therefore, our algorithm has a higher accuracy than the *waiting-and-matching* mechanism. And, because we only scan stream D once, the complexity of the algorithm is $O(N)$ for one objective pattern, where N is the length of D.

5 VIDEO ASSOCIATION CLASSIFICATION

To apply video associations in video indexing, we need to classify each association into a corresponding category (event) and use detected events to construct video indices. Some research efforts have addressed the problem of association rule classification, but little literature has been found on classifying sequential associations. We adopt the nearest neighbor search-based strategy as follows: We first mine associations from training videos. For each association, we manually go through the training data to evaluate what types of events associate with the appearance of this association. We count the number and the types of events from all appearances and select the event with the largest number to label the association. Accordingly, each association will receive one class label. For each association, $\{X\}$, in the test set, we calculate its distance with associations in the training set and the class label of the association in the training set which has the smallest distance with $\{X\}$ is used to label $\{X\}$. In the case that multiple associations have the same smallest distance with $\{X\}$, all their class labels are used to label $\{X\}$. To calculate the distance between sequential associations, we take the temporal order and the length of the associations into consideration and use the *Longest Common Subsequence* (*LCS*) [40] between two associations to evaluate the association distances.

Given two associations, assuming $\{X\}^1 = \{X_1, \ldots, X_P\}$ denotes the association with a length, P, and the other association is denoted by $\{X\}^2 = \{X_1, \ldots X_Q\}$ with length Q. For example,

$$\{X\}^1 = \{\mathrm{A, B, E, F, G}\}$$

and $\{X\}^2 = \{\mathrm{B, A, E, G, A, D}\}$. The Dynamic Programming [40] has $O(PQ)$ time complexity and space requirement to find the largest common subsequence between $\{X\}^1$ and $\{X\}^2$. [3] Then, the distance between $\{X\}^1$ and $\{X\}^2$ is defined by (10), where $|\mathrm{LCS}\{\{X\}^1, \{X\}^2\}|$ represents the length of the largest common subsequence:

$$SeqAssocD\{\{X\}^1, \{X\}^2\} = 1 - \frac{|LCS\{\{X\}^1, \{X\}^2\}|}{Min(P, Q)}, \quad (10)$$

Actually, this distance is determined by the maximal number of sequentially matched items between the

3. In the example above, there are two LCS subsequences $\mathrm{LCS}\{\{X\}^1, \{X\}^2\} = \{\{\mathrm{A, E, G}\}, \{\mathrm{B, E, G}\}\}$.

associations $\{X\}^1$ and $\{X\}^2$. The larger the number, the smaller their distance is.

6 Experimental Results

The results of an extensive performance analysis conducted to 1) evaluate the video processing techniques in Section 3, 2) evaluate the video association mining and association-based indexing algorithms in Sections 4 and 5, and 3) analyze the performance of our knowledge-based indexing framework are located in the Appendix which can be found on the Computer Society Digital Library at http://computer.org/tkde/archives.htm. Our algorithms were evaluated with eight basketball videos (NBA and NCAA) captured from ESPN and Fox and all commercials in the videos are removed.

7 Conclusions and Remarks

In this paper, we have proposed a solution for a new research area of video mining—video association mining. We have used video associations to construct a knowledge-based video indexing structure to support efficient video database management and access. We have introduced various techniques to extract visual and audio semantic cues and combined them into one hybrid stream by considering their original temporal order in the video. Consequently, the video data is transformed into a relational data set. We have employed a sequential multi level association mining strategy to mine associated video items and take them as video associations. We have adopted a scheme to classify associations into different categories, where each association can possibly indicate the happening of one type of event. The knowledge-based video indexing structure is accomplished by mining and classifying associations from video data. We have presented experimental results to demonstrate the performance of the proposed schemes. We believe we have explored a new research area to discover video knowledge for efficient video database management.

While the strategies presented in this paper are specific to basketball videos, mining associations for video knowledge exploration is an essential idea we want to convey here. From this point of view, further research could be conducted on the following aspects: 1) Extend the current framework to other domains and evaluate the performance of the video mining algorithm in environments containing more events. We believe the most promising domain is the surveillance video, where the routine vehicles in security areas normally comply with some associations like enter $\rightarrow$ stop $\rightarrow$ drop off $\rightarrow$ leave and a vehicle which does not comply with this association might be problematic and deserves further investigation. However, due to the inherent differences between different video domains (e.g., the concept of shot and video text do not exist in surveillance videos), we may need more efforts to analyze the video content details for association mining, e.g., extract trails and status of moving objects to characterize associations. 2) We have adopted various video processing techniques to explore visual and audio cues for association mining and it will inevitably incur information loss from the original video sequences to transferred symbolic streams; more studies are needed to address this issue in the mining activities. 3) The mining algorithms in this paper are mainly derived from the existing data mining schemes (with some extensions for video mining scenarios); extensive studies are needed to explore efficient mining algorithms which are unique for mining knowledge from video data.

Acknowledgment

The authors would like to thank the anonymous reviewers for their constructive comments on two earlier versions of this paper. This work was supported by the US Army Research Laboratory and the US Army Research Office under grant number DAAD19-02-1-0178, the US National Science Foundation under grants 0093116-IIS and 9972883-EIA, and the NSF under contract 69935010.

References

[1] H. Zhang, A. Kantankanhalli, and S. Smoliar, "Automatic Partitioning of Full-Motion Video," *ACM Multimedia Systems,* vol. 1, no. 1, pp. 10-28, 1993.

[2] A. Yoshitaka and T. Ichikawa, "A Survey on Content-Based Retrieval for Multimedia Databases," *IEEE Trans. Knowledge and Data Eng.,* vol. 11, no. 1, pp. 81-93, Jan./Feb. 1999.

[3] H. Jiang and A.K. Elmagarmid, "WVTDB—A Semantic Content-Based Video Database System on the World Wide Web," *IEEE Trans. Knowledge and Data Eng.,* vol. 10, no. 6, pp. 947-966, Nov./Dec. 1998.

[4] C. Snoek and M. Worring, "Multimodal Video Indexing: A Review of the State-of-the-Art," *Multimedia Tools and Applications,* to be published in 2005.

[5] F. Kokkoras, H. Jiang, I. Vlahavas, A. Elmagarmid, E. Houstis, and W. Aref, "Smart VideoText: A Video Data Model Based on Conceptual Graphs," *ACM/Springer Multimedia Systems,* vol. 8, no. 4, pp. 328-338, 2002.

[6] X. Zhu, J. Fan, W.G. Aref, and A.K. Elmagarmid, "ClassMiner: Mining Medical Video Content Structure and Events Towards Efficient Access and Scalable Skimming," *Proc. ACM SIGMOD Workshop,* pp. 9-16, 2002.

[7] X. Zhu, W. Aref, J. Fan, A. Catlin, and A. Elmagarmid, "Medical Video Mining for Efficient Database Indexing, Management and Access," *Proc. 19th Int'l Conf. Data Eng.,* pp. 569-580, 2003.

[8] Y. Matsuo, K. Shirahama, and K. Uehara, "Video Data Mining: Extracting Cinematic Rules from Movie," *Proc. Int'l Workshop Multimedia Data Management (MDM-KDD),* 2003.

[9] R.R. Wang and T.S. Huang, "A Framework of Human Motion Tracking and Event Detection for Video Indexing and Mining," *Proc. DIMACS Workshop Video Mining,* 2002.

[10] J. Oh and B. Bandi, "Multimedia Data Mining Framework for Raw Video Sequence," *Proc. Int'l Workshop Multimedia Data Management (MDM-KDD),* 2002.

[11] J. Pan and C. Faloutsos, "VideoCube: A Novel Tool for Video Mining and Classification," *Proc. Int'l Conf. Asian Digital Libraries (ICADL),* pp. 194-205, 2002.

[12] J. Pan and C. Faloutsos, "GeoPlot: Spatial Data Mining on Video Libraries," *Proc. Int'l Conf. Information and Knowledge Management,* pp. 405-412, 2002.

[13] X. Zhu and X. Wu, "Mining Video Association for Efficient Database Management," *Proc. Int'l Joint Conf. Artificial Intelligence,* pp. 1422-1424, 2003.

[14] X. Zhu and X. Wu, "Sequential Association Mining for Video Summarization," *Proc. IEEE Int'l Conf. Multimedia and Expo,* vol. 3, pp. 333-336, 2003.

[15] J. Fan, X. Zhu, and X. Lin, "Mining of Video Database," *Multimedia Data Mining,* 2002.

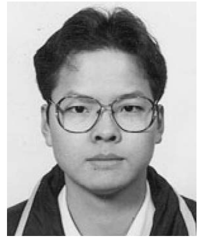
[16] L. Xie, S.-F. Chang, A. Divakaran, and H. Sun, "Unsupervised Mining of Statistical Temporal Structures in Video," *Video Mining,* A. Rosenfeld, D. Doremann, and D. Dementhon eds., Kluwer Academic, 2003.

[17] D. Wijesekera and D. Barbara, "Mining Cinematic Knowledge: Work in Progress," *Proc. Int'l Workshop Multimedia Data Management (MDM-KDD),* 2000.

[18] M. Windhouwer, R. Zwol, H. Blok, W. Jonker, M. Kersten, and P. Apers, "Content-Based Video Indexing for the Support of Digital Library Search," *Proc. Int'l Conf. Data Eng.,* pp. 494-495, 2002.

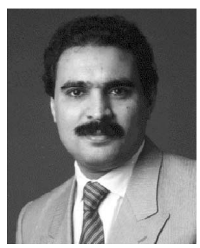
[19] S. Newsam, J. Tesic, L. Wang, and B.S. Manjunath, "Mining Images and Video," *Proc. DIMACS Workshop Video Mining,* 2002.

[20] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," *Proc. Very Large Data Bases Conf.*, pp. 487-499, 1994.
[21] R. Agrawal and R. Srikant, "Mining Sequential Patterns," *Proc. 11th Int'l Conf. Data Eng.*, 1995.
[22] J. Han and M. Kamber, *Data Mining: Concepts and Techniques.* Morgan Kaufmann, 2000.
[23] B. Thuraisingham, *Managing and Mining Multimedia Database.* CRC Press, 2001.
[24] O. Zaiane, J. Han, Z. Li, S. Chee, and J. Chiang, "MultimediaMiner: A System Prototype for Multimedia Data Mining," *Proc. ACM SIGMOD*, pp. 581-583, 1998.
[25] S. Nepal, U. Srinivasan, and G. Reynolds, "Automatic Detection of 'Goal' Segments in Basketball Videos," *Proc. Ninth ACM Multimedia Conf.*, pp. 261-269, 2001.
[26] L. Duan, M. Xu, T. Chua, Q. Tian, and C. Xu, "A Mid-Level Representation Framework for Semantic Sports Video Analysis," *Proc. of 11th ACM Multimedia Conf.*, pp. 33-44, 2003.
[27] J. Fan, W.G. Aref, A.K. Elmagarmid, M. Hacid, M. Marzouk, and X. Zhu, "MultiView: Multi-Level Video Content Representation and Retrieval," *J. Electronic Imaging*, vol. 10, no. 4, pp. 895-908, 2001.
[28] W. Zhou, A. Vellaikal, and C. Kuo, "Rule-Based Video Classification System for Basketball Video Indexing," *Proc. ACM Multimedia Workshops*, pp. 213-216, 2000.
[29] L. Xie, S. Chang, A. Divakaran, and H. Sun, "Structure Analysis of Soccer Video with Hidden Markov Models," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, 2002.
[30] W. Wolf, "Key Frame Selection by Motion Analysis," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 1228-1231, 1996.
[31] H. Tamura, S. Mori, and T. Yamawaki, "Texture Features Corresponding to Visual Perception," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 8, no. 6, pp. 460-473, 1978.
[32] S. Horowitz and T. Pavlidis, "Picture Segmentation by a Directed Split-and-Merge Procedure," *Proc. Int'l Joint Conf. Pattern Recognition*, pp. 424-433, 1974.
[33] WOCAR Engine 2.5, http://ccambien.free.fr/wocar/, 2004.
[34] A. Jain and B. Yu, "Automatic Text Location in Images and Video Frames," *Pattern Recognition*, vol. 31, no. 12, pp. 2055-2076, 1998.
[35] X. Zhu, A.K. Elmagarmid, X. Xue, L. Wu, and A. Catlin, "InsightVide: Towards Hierarchical Video Content Organization for Efficient Browsing, Summarization, and Retrieval," *IEEE Trans. Multimedia*, 2004.
[36] T. Oates and P. Cohen, "Searching for Structure in Multiple Streams of Data," *Proc. 13th Int'l Conf. Machine Learning*, pp. 346-354, 1996.
[37] J. Han, G. Dong, and Y. Yin, "Efficient Mining Partial Periodic Patterns in Time Series Database," *Proc. Int'l Conf. Data Eng.*, pp. 106-115, 1999.
[38] R. Srikant and R. Agrawal, "Mining Generalized Association Rules," *Proc. 21th Very Large Data Bases Conf.*, 1995.
[39] R. Gwadera, M. Atallah, and W. Szpankowski, "Reliable Detection of Episodes in Event Sequences," *Proc. Third Int'l Conf. Data Mining*, pp. 67-74, 2003.
[40] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms.* MIT Press, 2001.
[41] B. Cui, B. Ooi, J. Su, and K. Tan, "Contorting High Dimensional Data for Efficient Main Memory Processing," *Proc. SIGMOD Conf.*, pp. 479-490, 2003.
[42] A. Guttman, "R-Trees: A Dynamic Index Structure for Spatial Searching," *Proc. SIGMOD Conf.*, pp. 47-57, 1984.
[43] N. Katayama and S. Satoh, "The SR-Tree: An Index Structure for High-Dimensional Nearest Neighbor Queries," *Proc. SIGMOD Conf.*, pp. 369-380, 1997.
[44] W. Hsu, J. Dai, and M. Lee, "Mining Viewpoint Patterns in Image Databases," *Proc. SIGKDD*, pp. 553-558, 2003.
[45] H. Mannila, H. Toivonen, and A. Verkamo, "Discovery of Frequent Episodes in Event Sequences," *Data Mining and Knowledge Discovery*, vol. 1, no. 3, pp. 259-289, 1997.
[46] R. Srikant and R. Agrawal, "Mining Sequential Patterns: Generalizations and Performance Improvements," *Proc. Fifth Int'l Conf. Extending Database Technology (EDBT)*, 1996.

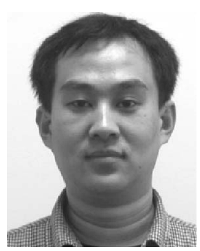


Xingquan Zhu received the PhD degree in computer science from Fudan University, Shanghai, China, in 2001. He is a research assistant professor in the Department of Computer Science at the University of Vermont. He spent four months with Microsoft Research Asia, Beijing, China, where he was working on content-based image retrieval with relevance feedback. From 2001 to 2002, he was a postdoctoral associate in the Department of Computer Science, Purdue University, West Lafayette, Indiana. His research interests include data mining, machine learning, data quality, multimedia systems, and information retrieval. Since 2000, Dr. Zhu has published extensively, including more than 40 refereed papers in various journals and conference proceedings. He is a member of the IEEE.

Xindong Wu received the PhD degree in artificial intelligence from the University of Edinburgh, Britain. He is a professor and chair of the Department of Computer Science at the University of Vermont. His research interests include data mining, knowledge-based systems, and Web information exploration. He has published extensively in these areas in various journals and conferences. He has been an invited/keynote speaker at six international conferences. Dr. Wu is the Editor-in-Chief of the *IEEE Transactions on Knowledge and Data Engineering* and an honorary Editor-in-Chief (starting 1 January 2005) of *Knowledge and Information Systems* (a peer-reviewed archival journal published by Springer-Verlag). He is the winner of the 2004 ACM SIGKDD Service. He is a senior member of the IEEE.



Ahmed K. Elmagarmid received the BS degree in computer science from the University of Dayton and the MS and PhD degrees from The Ohio State University in 1977, 1981, and 1985, respectively. He received a Presidential Young Investigator award from the US National Science Foundation and the distinguished alumni awards from Ohio State University and the University of Dayton in 1988, 1993, and 1995, respectively. Professor Elmagarmid is the editor-in-chief of *Distributed and Parallel Databases: An International Journal* and of the book series on Advances in Database Systems. He serves or has served on numerous editorial boards. His research interests are in the areas of video databases, multidatabases, data quality, and their applications in telemedicine and digital government. He is the author of several books in databases and multimedia. He has been with the Department of Computer Science at Purdue University since 1988. He was chief scientist for Hewlett-Packard from 2001 to 2003 while on leave from Purdue. He is a senior member of the IEEE.



Zhe Feng received the PhD degree in computer science from Fudan University, Shanghai, China, in 2004. He is now a research assistant in the Department of Computer Science, Fudan University. His research interests include audio/video content analysis and multimedia information retrieval.

Lide Wu graduated from the Department of Mathematics at Fudan University, and was working there until 1975. Since then, he has been with the Department of Computer Science at the same university and is now a chair professor there. His research interests include image processing, video processing, computer vision, pattern recognition, and Chinese text processing. He is the author or coauthor of more than 10 books and more than 200 papers. He is vice chairman of technical committee of AI & PR of Chinese Computer Society, a senior member of the IEEE, and a member of the ACM and the New York Academy of Sciences.