Chapter 9.2 A Unified Framework for Video Summarization, Browsing and Retrieval

Ziyou Xiong[†], Yong Rui[♮], Regunathan Radhakrishnan[‡], Ajay Divakaran[‡], Thomas S. Huang[†]

[†]Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA
E-mail:{zxiong, huang}@ifp.uiuc.edu

[♮]Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA
E-mail:yongrui@microsoft.com

[‡]Mitsubishi Electric Research Labs
201, Broadway, Cambridge, MA 02139, USA
E-mail:{regu, ajayd}@merl.com

1 Introduction

Video content can be accessed by using either a top-down approach or a bottom-up approach [1, 2, 3, 4]. The top-down approach, i.e. video browsing, is useful when we need to get an "essence" of the content. The bottom-up approach, i.e. video retrieval, is useful when we know exactly what we are looking for in the content, as shown in Fig. 1. In video summarization, what "essence" the summary should capture depends on whether the content is scripted or not. Since scripted content, such as news, drama & movie, is carefully structured as a sequence of semantic units, one can get its essence by enabling a traversal through representative items from these semantic units. Hence, Table of Contents (ToC) based video browsing caters to summarization of scripted content. For instance, a news video composed of a sequence of stories can be summarized/browsed using a key-frame representation for each of the shots in a story. However, summarization of unscripted content, such as surveillance & sports), requires a "highlights" extraction framework that only captures remarkable events that constitute the summary.

Considerable progress has been made in multimodal analysis, video representation, summarization, browsing and retrieval, which are the five fundamental bases for accessing video content. The first three bases focus on meta-data generation & organization while the last two focus on meta-data consumption. *Multimodal analysis* deals with the *signal processing* part of the video system, including shot boundary detection, key frame extraction, key object detection, audio analysis, closed caption analysis etc. *Video representation* is concerned with the *structure* of the video.

Again, it is useful to have different representations for scripted and unscripted content. An example of a video representation for scripted content, is the tree structured key frame hierarchy [5, 3]. Built on top of the video representation, *video summarization*, either based on ToC generation or highlights extraction, deals with how to use the representation structure to provide the viewers top-down access using the summary for video browsing. Finally, *video retrieval* is concerned with retrieving specific video objects. The relationship between these five bases is illustrated in Fig. 1.

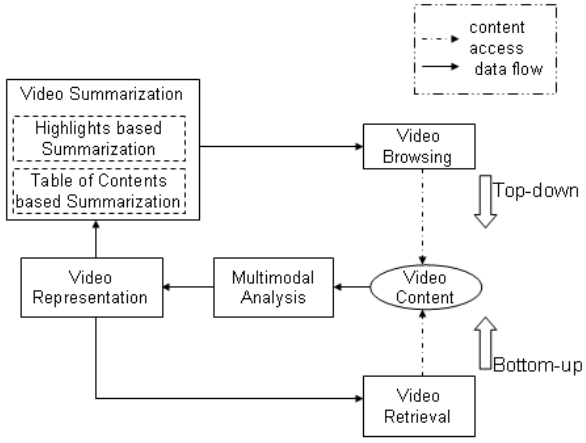

Figure 1: Relations between the five research areas

As seen in Fig. 1, video browsing and retrieval *directly* support users' access to the video content. For accessing a temporal medium, such as a video clip, summarization, browsing and retrieval are equally important. As mentioned earlier, browsing enabled through summarization helps a user to quickly grasp the global picture of the data, while retrieval helps a user to find the results of a specific query.

An analogy explains this argument. How does a reader efficiently access the content of a 1000-page book? Without reading the whole book, he can first go to the book's Table-of-Contents (ToC) to find which chapters or sections suit his needs. If he has specific questions (queries) in mind, such as finding a term or a key word, he can go to the Index at the end of the book and find the corresponding book sections addressing that question. On the other hand, how does a reader efficiently access the content of a 100-page magazine? Without reading the whole magazine, he can either directly go to the featured articles listed on the front page or use the ToC to find which article suits his needs. In short, the book's ToC helps a reader *browse*, and the book's index helps a reader *retrieve*. Similarly, the magazine's featured articles also help the reader *browse* through the highlights. All these three aspects are equally important in helping users access the content of the book or the magazine. For today's video content, techniques are urgently needed for automatically (or semi-automatically) constructing video ToC, video Highlights and video Indices to facilitate summarization, browsing and retrieval.

A great degree of power and flexibility can be achieved by simultaneously designing the video access components (ToC, Highlights and Index) using a unified framework. For a long and continuous stream of data, such as video, a "back and forth" mechanism between summarization and retrieval is crucial.

The goals of this chapter are to develop novel techniques for constructing the video ToC, video Highlights and video Index as well as how to integrate them into a unified framework. The rest of the chapter is organized as follows. In Section 2, important video terms are first introduced. We review video analysis, representation, summarization, and retrieval in Sections 3 to 5, respectively. In Section 6 we describe in detail a unified framework for video summarization and retrieval. Algorithms as well as experimental results on real-world video clips are presented. Conclusions and future research directions are summarized in Section 7.

2 Terminology

Before we go into the details of the discussion, it will be beneficial to first introduce some important terms used in the digital video research field.

- *Scripted/Unscripted Content*: A video that is carefully produced according to a script or plan that is later edited, compiled and distributed for consumption is referred to as scripted content. News videos, dramas & movies are examples of scripted content. Video content that is not scripted is then referred to as unscripted. In unscripted content, such as surveillance video, the events happen spontaneously. One can think of varying degrees of "scripted-ness" & "unscripted-ness" from movie content to surveillance content.

- *Video shot*: is a consecutive sequence of frames recorded from a single camera. It is the building block of video streams.

- *Key frame*: is the frame which represents the salient visual content of a shot. Depending on the complexity of the content of the shot, one or more key frames can be extracted.

- *Video scene*: is defined as a collection of semantically related and temporally adjacent shots, depicting and conveying a high-level concept or story. While shots are marked by physical boundaries, scenes are marked by semantic boundaries [1].

- *Video group*: is an intermediate entity between the physical shots and semantic scenes and serves as the bridge between the two. Examples of groups are temporally adjacent shots [5] or visually similar shots [3].

- *Play and Break*: is the first level of semantic segmentation in sports video and surveillance video. In sports video (e.g soccer, baseball, golf), a game is in play when the ball is in the field and the game is going on; break, or out of play, is the complement set, i.e., whenever "the ball has completely crossed the goal line or touch line, whether on the ground or in the air" or "the game has been halted by the referee"[6]. In surveillance video, a play is a period in which there is some activity in the scene.

- *Audio Marker*: is a contiguous sequence of audio frames representing a key audio class that is indicative of the events of interest in the video. An example of an audio marker for sports

[1] Some of the early literature in video parsing misused the phrase *scene change detection* for *shot boundary detection*. To avoid any later confusion, we will use *shot boundary detection* to mean the detection of physical shot boundaries while using *scene boundary detection* to mean the detection of semantic scene boundaries.

video can be the audience reaction sound (cheering & applause) or commentator's excited speech.

- *Video Marker*: is a contiguous sequence of video frames containing a key video object that is indicative of the events of interest in the video. An example of a video marker for baseball videos is the video segment containing the squatting catcher at the beginning of every pitch.
- *Highlight Candidate*: is a video segment that is likely to be remarkable and can be identified using the video and audio markers.
- *Highlight Group*: is a cluster of highlight candidates.

In summary, scripted video data can be structured into a hierarchy consisting of five levels: video, scene, group, shot, and key frame, which increase in granularity from top to bottom [4] (see Fig. 2). Similarly, the unscripted video data can be structured into a hierarchy of four levels: play/break, audio-visual markers, highlight candidates, highlight groups, which increase in semantic level from bottom to top (see Fig. 3).

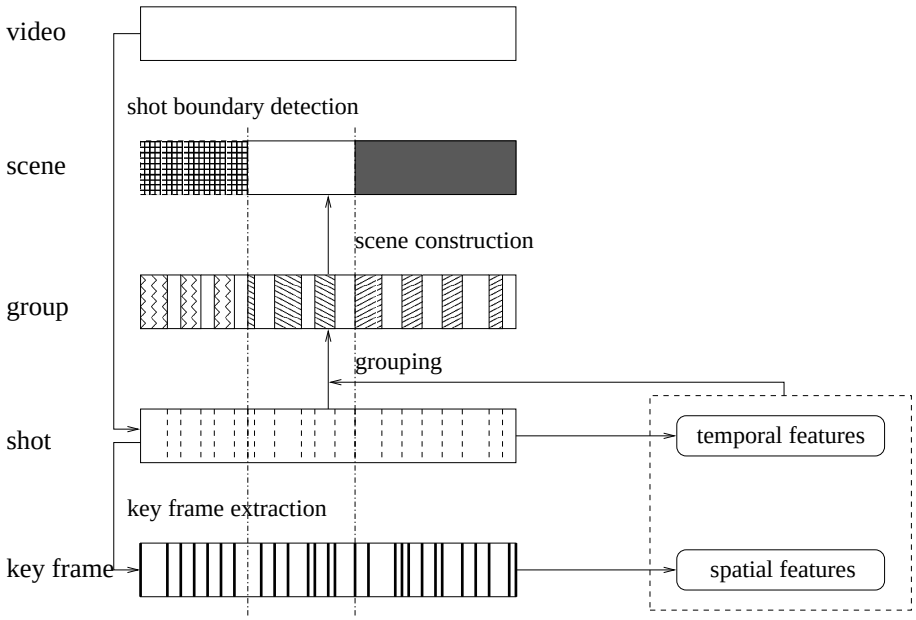

Figure 2: A hierarchical video representation for scripted content

3 Video Analysis

As can be seen from Fig. 1, multimodal analysis is the basis for later video processing. It includes *shot boundary detection* and *key frame extraction* for scripted content. For unscripted content, it includes *play/break segmentation*, *audio marker detection* & *visual marker detection*.

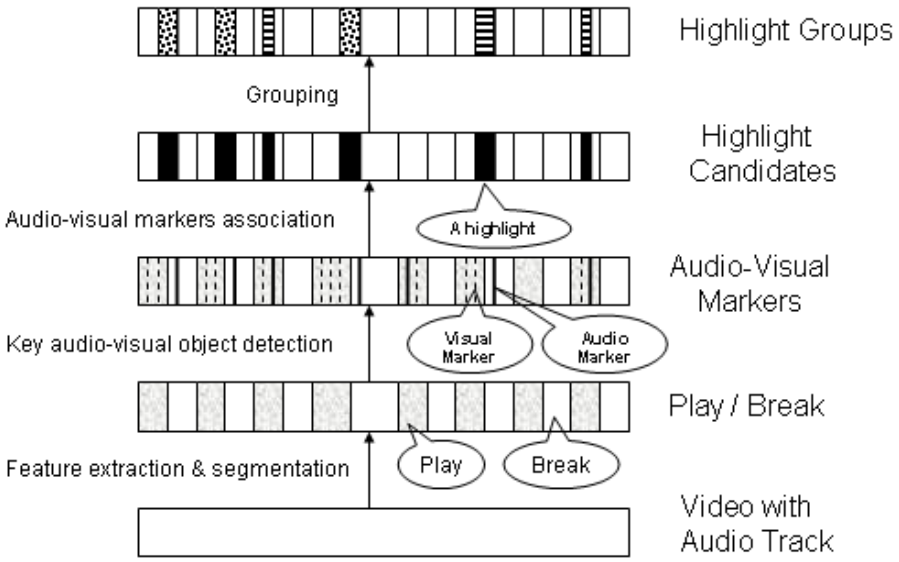

Figure 3: A hierarchical video representation for unscripted content

3.1 Shot Boundary Detection

It is not efficient (sometimes not even possible) to process a video clip as a whole. It is beneficial to first decompose the video clip into shots and do the signal processing at the shot level.

In general, automatic shot boundary detection techniques can be classified into five categories: *pixel-based*, *statistics-based*, *transform-based*, *feature-based*, and *histogram-based*. Pixel-based approaches use pixel-wise intensity difference to mark shot boundaries [1, 7]. However, they are highly sensitive to noise. To overcome this problem, Kasturi and Jain propose to use intensity statistics (mean and standard deviation) as shot boundary detection measures [8]. In order to achieve faster processing, Arman, Hsu and Chiu propose to use the compressed DCT coefficients (e.g., MPEG data) as the boundary measure [9]. Other transform-based shot boundary detection approaches make use of motion vectors, which are already embedded in the MPEG stream [10, 11]. Zabih et al. address the problem from another angle. Edge features are first extracted from each frame. Shot boundaries are then detected by finding sudden edge changes [12]. So far, the histogram-based approach is the most popular. Instead of using pixel intensities directly, the histogram-based approach uses histograms of the pixel intensities as the measure. Several researchers claim that it achieves a good tradeoff between accuracy and speed [1]. Representatives of this approach are [13, 14, 1, 15, 16]. More recent work has been based on clustering and post-filtering [17], which achieves fairly high accuracy without producing many false positives. Two comprehensive comparisons of shot boundary detection techniques are presented in [18, 19].

3.2 Key Frame Extraction

After the shot boundaries are detected, corresponding key frames can then be extracted. Simple approaches may just extract the first and last frames of each shot as the key frames [16]. More sophisticated key frame extraction techniques are based on visual content complexity indicators [20], shot activity indicators [21], and shot motion indicators [22][23].

The following three analysis steps mainly cater to the analysis of unscripted content.

3.3 Play/Break Segmentation

Since unscripted content has short periods of activity (plays) between periods of inactivity (breaks), it is useful to first segment the whole content into these units. This helps in reducing the amount of content to be analyzed for subsequent processing that looks for highlight segments within plays. Play/break segmentation for sports, both in an unsupervised and supervised manner using low-level features, has been reported in [24]. Play/break segmentation in surveillance has been reported using adaptive background subtraction techniques that identify periods of object activity from the whole content [25].

3.4 Audio Marker Detection

Audio markers are key audio classes that are indicative of the events of interest in unscripted content. In our previous work on sports, audience reaction & commentator's excited speech are classes that have been shown to be useful markers [26] [27]. Nepal et al. [28] detect basketball "goal" based on crowd cheers from the audio signal using energy thresholds. Another example of an audio marker, consisting of keywords such as "touchdown" or "fumble", has been reported in [29].

3.5 Video Marker Detection

Visual markers are key video objects that are indicative of the events of interest in unscripted content. Some examples of useful and detectable visual markers are: "the squatting baseball catcher pose" for baseball, "the goal post" for soccer etc. Kawashima et al. [30] detect bat-swings as visual markers using visual features. Gong et al. [31] detect and track visual markers such as the soccer court, the ball, the players, and the motion patterns.

4 Video Representation

Considering that each video frame is a 2D object and the temporal axis makes up the third dimension, a video stream spans a 3D space. Video representation is the *mapping* from the 3D space to the 2D view screen. Different mapping functions characterize different video representation techniques.

4.1 Video Representation for Scripted Content

Using an analysis framework that can detect shots, key frames and scenes, it is possible to come-up with the following representations for scripted content.

4.1.1 Representation based on Sequential Key Frames

After obtaining shots and key frames, an obvious and simple video representation is to sequentially lay out the key frames of the video, from top to bottom and from left to right. This simple technique works well when there are few key frames. When the video clip is long, this technique does not scale, since it does not capture the embedded information within the video clip, except for time.

4.1.2 Representation based on Groups

To obtain a more meaningful video representation when the video is long, related shots are merged into groups [5, 3]. In [5], Zhang et al. divide the entire video stream into multiple video segments, each of which contains an equal number of consecutive shots. Each segment is further divided into sub-segments; thus constructing a tree structured video representation. In [3], Zhong et al. proposed a cluster-based video hierarchy, in which the shots are clustered based on their visual content. This method again constructs a tree structured video representation.

4.1.3 Representation based on Scenes

To provide the user with better access to the video, the construction of a video representation at the semantic level is needed [4, 2]. It is not uncommon for a modern movie to contain a few thousand shots and key frames. This is evidenced in [32] – there are 300 shots in a 15-minute video segment of the movie "Terminator 2 - Judgment Day" and the movie lasts 139 minutes. Because of the large number of key frames, a simple 1D sequential presentation of key frames for the underlying video (or even a tree structured layout at the group level) is almost meaningless. More importantly, people watch the video by its semantic scenes rather than the physical shots or key frames. While *shot* is the building block of a video, it is *scene* that conveys the semantic meaning of the video to the viewers. The discontinuity of shots is overwhelmed by the continuity of a scene [2]. Video ToC construction at the scene level is thus of fundamental importance to video browsing and retrieval. In [2], a scene transition graph (STG) of video representation is proposed and constructed. The video sequence is first segmented into shots. Shots are then clustered by using *time-constrained clustering*. The STG is then constructed based on the time flow of the clusters.

4.1.4 Representation based on Video Mosaics

Instead of representing the video structure based on the video-scene-group-shot-frame hierarchy as discussed above, this approach takes a different perspective [33]. The mixed information within a shot is decomposed into three components:

- *Extended spatial information*: this captures the appearance of the entire background imaged in the shot, and is represented in the form of a few mosaic images.
- *Extended temporal information*: this captures the motion of independently moving objects in the form of their trajectories.
- *Geometric information*: this captures the geometric transformations that are induced by the motion of the camera.

4.2 Video Representation for Unscripted Content

Highlights extraction from unscripted content requires a different representation from the one that supports browsing of scripted content. This is because shot detection is known to be unreliable for unscripted content. For example, in soccer video, visual features are so similar over a long period of time that almost all the frames within it, may be grouped as a single shot. However, there might be multiple semantic units within the same period such as attacks on the goal, counter attacks in the mid-field, etc. Furthermore, the representation of unscripted content should emphasize detection of remarkable events to support highlights extraction while the representation for scripted content does not fully support the notion of an event being remarkable compared to others.

For unscripted content, using an analysis framework that can detect plays & specific audio and visual markers, it is possible to come up with the following representations.

4.2.1 Representation based on Play/Break Segmentation

As mentioned earlier, play/break segmentation using low-level features gives a segmentation of the content at the lowest semantic level. By representing a key frame from each of the detected play segments, one can enable the end user to select just the play segments.

4.2.2 Representation based on Audio-Visual Markers

The detection of audio & visual markers enables a representation that is at a higher semantic level than play/break representation is. Since the detected markers are indicative of the events of interest, the user can use either or both of them to browse the content based on this representation.

4.2.3 Representation based on Highlight Candidates

Association of an audio marker with a video marker enables detection of highlight candidates which are at a higher semantic level. Such a fusion of complementary cues from audio & video helps eliminate false alarms in either of the marker detectors. Segments in the vicinity of a video marker and an associated audio marker give access to the highlight candidates for the end-user. For instance, if the baseball catcher pose (visual marker) is associated with an audience reaction segment (audio marker) that follows it closely, the corresponding segment is highly likely to be remarkable or interesting.

4.2.4 Representation based on Highlight Groups

Grouping of highlight candidates would give a finer resolution representation of the highlight candidates. For example, golf swings and putts share the same audio markers (audience applause and cheering) and visual markers (golfers bending to hit the ball). A representation based on highlight groups, supports the task of retrieving finer events such as "golf swings only" or "golf putts only".

5 Video Browsing and Retrieval

These two functionalities are the ultimate goals of a video access system, and they are closely related to (and built on top of) video representations. The representation techniques for scripted content discussed above are suitable for browsing through ToC based summarization while the last can be used in video retrieval. On the other hand, the representation techniques for unscripted content are suitable for browsing through highlights based summarization.

5.1 Video Browsing using ToC based Summary

For "Representation based on Sequential Key Frames", browsing is obviously sequential browsing, scanning from the top-left key frame to the bottom-right key frame. For "Representation based on Groups", a hierarchical browsing is supported [5, 3]. At the coarse level, only the main themes are displayed. Once the user determines which theme he is interested in, he can then go to the finer level of the theme. This refinement process can go on until the leaf level. For the STG representation, a major characteristic is its indication of time flow embedded within the representation. By following the time flow, the viewer can browse through the video clip.

5.2 Video Browsing using Highlights based Summary

For "Representation based on Play/Break Segmentation", browsing is also sequential, enabling a scan of all the play segments from the beginning of the video to the end. "Representation based on Audio-Visual Markers" supports queries such as "find me video segments that contain the soccer goal post in the left-half field", "find me video segments that have the audience applause sound" or "find me video segments that contain the squatting baseball catcher". "Representation based on Highlight Candidates" supports queries such as "find me video segments where a golfer has a good hit" or "find me video segments where there is a soccer goal attempt". Note that "a golfer has a good hit" is represented by the detection of the golfer hitting the ball followed by the detection of applause from the audience. Similarly, that "there is a soccer goal attempt" is represented by the detection of the soccer goal post followed by the detection of long and loud audience cheering. "Representation based on Highlight Groups" supports more detailed queries than the previous representation. These queries include "find me video segments where a golfer has a good swing ", "find me video segments where a golfer has a good putt", or "find me video segments where there is a good soccer corner kick" etc.

5.3 Video Retrieval

As discussed in Section 1, the ToC, Highlights and Index are all equally important for accessing the video content. Unlike the other video representations, the mosaic representation is especially suitable for video retrieval. Three components: moving objects, backgrounds, and camera motions, are perfect candidates for a video Index. After constructing such a video index, queries such as "find me a car moving like this", "find me a conference room having that environment", etc. can be effectively supported.

6 Proposed Framework

As we have reviewed in the previous sections, considerable progress has been made in each of the areas of video analysis, representation, browsing, and retrieval. However, so far, the interaction among these components is still limited and we still lack a unified framework to glue them together. This is especially crucial for video, given that the video medium is characteristically long and unstructured. In this section, we will explore the synergy between video browsing and retrieval.

6.1 Video Browsing

Among the many possible video representations, the "Scene Based Representation" is probably the most effective for meaningful video browsing [4, 2]. We have proposed a scene-based video ToC representation in [4]. In this representation, a video clip is structured into the scene-group-shot-frame hierarchy (see Fig. 2), which then serves as the basis for the ToC construction. This ToC frees the viewer from doing tedious "fast forward" and "rewind", and provides the viewer with non-linear access to the video content. Fig. 4 and Fig. 5 illustrate the browsing process, enabled by the video ToC. Fig. 4 shows a condensed ToC for a video clip, as we normally have in a long book. By looking at the representative frames and text annotation, the viewer can determine which particular portion of the video clip he is interested in. Then, the viewer can further expand the ToC into more detailed levels, such as groups and shots. The expanded ToC is illustrated in Fig. 5. Clicking on the "Display" button will display the specific portion that is of interest to the viewer, without viewing the entire video.

The algorithm is described below. To learn details, interested readers are referred to [34].

[**Main procedure**]

- Input: Video shot sequence, $S = \{shot\ 0, ..., shot\ i\}$.
- Output: Video structure in terms of *scene*, *group*, and *shot*.
- Procedure:
  1. Initialization: assign shot 0 to group 0 and scene 0; initialize the group counter $numGroups = 1$; initialize the scene counter $numScenes = 1$.
  2. If S is empty, quit; otherwise get the next shot. Denote this shot as shot i.
  3. Test if shot i can be merged to an existing group:

Figure 4: The condensed ToC

(a) Compute the similarities between the current shot and existing groups: Call $findGroupSim()$.

(b) Find the maximum group similarity:

$$maxGroupSim_i = \max_g \quad GroupSim_{i,g} \quad , g = 1, ..., numGroups \tag{1}$$

where $GroupSim_{i,g}$ is the similarity between shot i and group g. Let the group of the maximum similarity be group g_{max}.

(c) Test if this shot can be merged into an existing group:
If $maxGroupSim_i > groupThreshold$, where $groupThreshold$ is a predefined threshold:

i. Merge shot i to group g_{max}.

ii. Update the video structure: Call $updateGroupScene()$.

iii. Goto Step 2.

otherwise:

i. Create a new group containing a single shot i. Let this group be group j.

ii. Set $numGroups = numGroups + 1$.

4. Test if shot i can be merged to an existing scene:

(a) Calculate the similarities between the current shot i and existing scenes: Call $findSceneSim()$.

(b) Find the maximum scene similarity:

$$maxSceneSim_i = \max_s \quad SceneSim_{i,s} \quad , s = 1, ..., numScenes \tag{2}$$

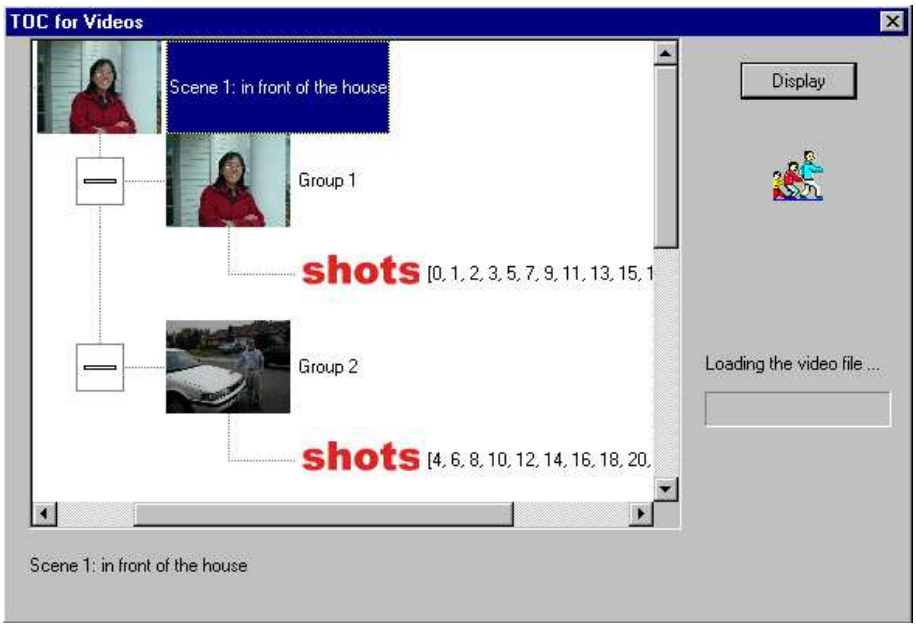

Figure 5: The expanded ToC

where $SceneSim_{i,s}$ is the similarity between shot i and scene s. Let the scene of the maximum similarity be scene s_{max}.

(c) Test if shot i can be merged into an existing scene:
If $maxSceneSim_i > sceneThreshold$, where $sceneThreshold$ is a predefined threshold:
   i. Merge shot i to scene s_{max}.
   ii. Update the video structure: Call $updateScene()$.
   otherwise:
   i. Create a new scene containing a single shot i and a single group j.
   ii. Set $numScenes = numScenes + 1$.

5. Goto Step 2.

[**findGroupSim**]

- Input: Current shot and group structure.
- Output: Similarity between current shot and existing groups.
- Procedure:
  1. Denote current shot as shot i.
  2. Calculate the similarities between shot i and existing groups:

$$GroupSim_{i,g} = ShotSim_{i,g_{last}} \quad , g = 1, ..., numGroups \tag{3}$$

where $ShotSim_{i,j}$ is the similarity between shots i and j; and g is the index for groups and g_{last} is the last (most recent) shot in group g. That is, the similarity between current

shot and a group is the similarity between the current shot and the most recent shot in the group. The most recent shot is chosen to represent the whole group because all the shots in the same group are visually similar and the most recent shot has the largest *temporal attraction* to the current shot.

3. Return.

[**findSceneSim**]

- Input: The current shot, group structure and scene structure.
- Output: Similarity between the current shot and existing scenes.
- Procedure:
  1. Denote the current shot as shot i.
  2. Calculate the similarity between shot i and existing scenes:

$$SceneSim_{i,s} = \frac{1}{numGroups_s} \sum_{g}^{numGroups_s} GroupSim_{i,g} \tag{4}$$

  where s is the index for scenes; $numGroups_s$ is the number of groups in scene s; and $GroupSim_{i,g}$ is the similarity between current shot i and g^{th} group in scene s. That is, the similarity between the current shot and a scene is the average of similarities between the current shot and all the groups in the scene.

  3. Return.

[**updateGroupScene**]

- Input: Current shot, group structure, and scene structure.
- Output: An updated version of group structure and scene structure.
- Procedure:
  1. Denote current shot as shot i and the group having the largest similarity to shot i as group g_{max}. That is, shot i belongs to group g_{max}.
  2. Define two shots, *top* and *bottom*, where *top* is the second most recent shot in group g_{max} and *bottom* is the most recent shot in group g_{max} (i.e., current shot).
  3. For any group g, if any of its shots (*shot* g_j) satisfies the following condition

$$top < shot\ g_j < bottom \tag{5}$$

  merge the scene that group g belongs to into the scene that group g_{max} belongs to. That is, if a scene contains a shot which is interlaced with the current scene, merge the two scenes. This is illustrated in Fig. 4 (shot i = shot 4, $g_{max} = 0$, $g = 1$, *top* = shot 1, and *bottom* = shot 4).

  4. Return.

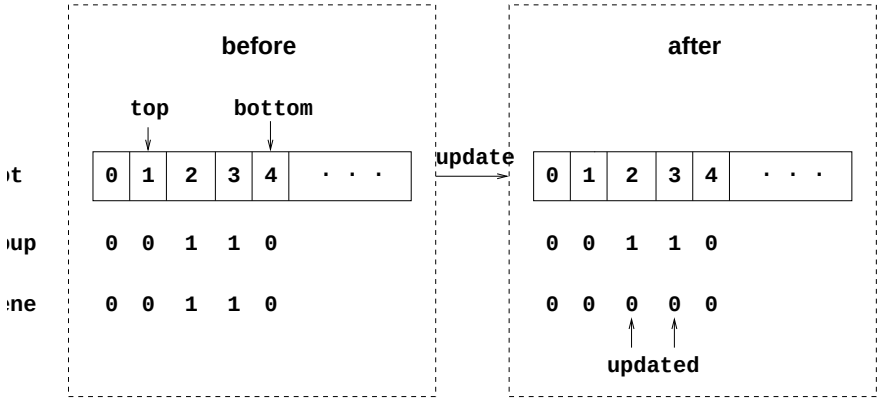

Figure 6: Merging scene 1 to scene 0

[**updateScene**]

- Input: Current shot, group structure, and scene structure.
- Output: An updated version of scene structure.
- Procedure:
  1. Denote current shot as shot i and the scene having the largest similarity to shot i as scene s_{max}. That is, shot i belongs to scene s_{max}.
  2. Define two shots, *top* and *bottom*, where *top* is the second most recent shot in scene s_{max} and *bottom* is the current shot in scene s_{max} (i.e., current shot).
  3. For any scene s, if any of its shots (*shot* s_j) satisfies the following condition

$$top < shot\ s_j < bottom \tag{6}$$

     merge scene s into scene s_{max}. That is, if a scene contains a shot which is interlaced with the current scene, merge the two scenes.
  4. Return.

Extensive experiments using real-world video clips have been carried out. The results are summarized in Table 1 [4], where "ds" (detected scenes) denotes the number of scenes detected by the algorithm; "fn" (false negatives) indicates the number of scenes missed by the algorithm; and "fp" (false positives) indicates the number of scenes detected by the algorithm, although they are considered scenes by humans.

Some observations can be summarized as follows:

- The proposed scene construction approach achieves reasonably good results in most of the movie types.
- The approach achieves better performance in "slow" movies than in "fast" movies. This follows since in the "fast" movies, the visual content is normally more complex and more difficult to capture. We are currently integrating closed-captioning information into the framework to enhance the accuracy of the scene structure construction.

Table 1. Scene structure construction results.

movie name	frames	shots	groups	ds	fn	fp
Movie1	21717	133	16	5	0	0
Movie2	27951	186	25	7	0	1
Movie3	14293	86	12	6	1	1
Movie4	35817	195	28	10	1	2
Movie5	18362	77	10	6	0	0
Movie6	23260	390	79	24	1	10
Movie7	35154	329	46	14	1	2

- The proposed approach seldom misses a scene boundary, but tends to over-segment the video. That is, "false positives" outnumber "false negatives". This situation is expected for most of the automated video analysis approaches and has also been observed by other researchers [32, 2].

A possible remedy to the drawbacks in the stated segmentation scheme can be eliminated by introducing the audio component of video also into the scene change scheme [35] [36]. This introduces the concept of a documentary scene, which is inferred from the National institute of standards and technology(NIST) produced documentary videos. It uses an audio-visual score value (which is a weighted combination of an audio score and a video score) to divide the video into a set of documentary scenes. The audio score and the visual score are generated by procedures evolved out of the observations that we make on the video data.

This scheme is illustrated in Fig. 7. Given a video, the proposed approach first generates a visual pattern and an audio pattern respectively based on similarity measures. It also makes use of the similarity in image background of the video frames for scene analysis[37]. The information collected from visual, audio and background based scene analysis will be integrated for audio-visual fusion. The scene change detector is composed of two main steps: adaptive scheme and redundancy check. In the adaptive scheme, the audio-visual score within an interval is first evaluated. This interval will be adaptively expanded or shrunk until a local maximum is found. In redundancy check, we eliminate the redundant scenes by a merging process.

6.2 Video Highlights Extraction

In this section, we describe our proposed approach for highlights extraction from "unscripted" content. We show the framework's effectiveness in three different sports namely soccer, baseball and golf. Our proposed framework can be summarized in Fig. 8. There are 4 major components in Fig. 8. We describe them one by one in the following.

6.2.1 Audio Marker Detection

Broadcast sports content usually includes audience reactions to the interesting moments of the games. Audience reaction classes including applause, cheering, and commentator's excited speech can serve as audio markers. We have developed classification schemes that can achieve very high

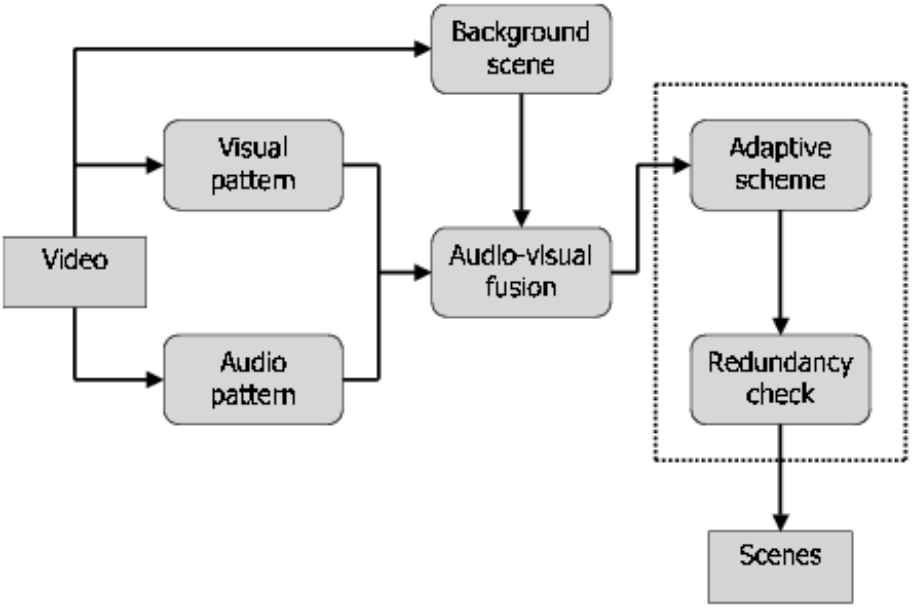

Figure 7: Proposed approach for detecting documentary scenes.

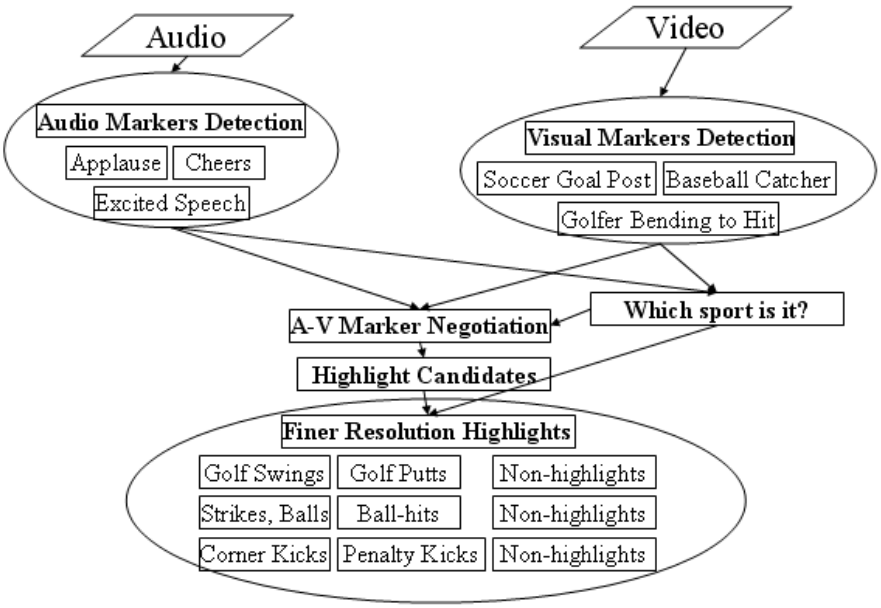

Figure 8: Proposed approach for sports highlights extraction.

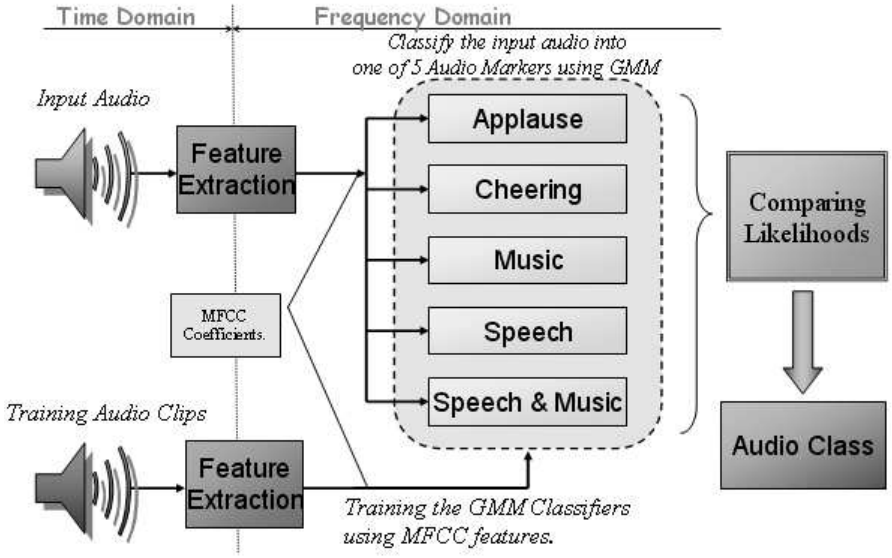

Figure 9: Audio markers for sports highlights extraction.

recognition accuracy on these key audio classes [26]. Fig. 9 shows our proposed unified audio marker detection framework for sports highlights extraction.

6.2.2 Visual Marker Detection

As defined earlier, visual markers are key visual objects that are indicative of the interesting segments. Fig. 10 shows examples of some visual markers for three different games. For baseball games, we want to detect the pattern in which the catcher squats waiting for the pitcher to pitch the ball; for golf games, we want to detect the players bending to hit the golf ball; for soccer, we want to detect the appearance of the goal post. Correct detection of these key visual objects can eliminate the majority of the video content that is not in the vicinity of the interesting segments. For the goal of one general framework for all three sports, we use the following processing strategy: for the unknown sports content, we detect whether there are baseball catchers, or golfers bending to hit the ball, or soccer goal posts. The detection results can enable us to decide which sport (baseball, golf or soccer) it is.

We use an object detection algorithm such as Viola and Jones's [38] to detect these visual markers. We have collected more than 8000 thumbnail images for positive examples. We have also collected more than 1000 images of size 352×240 from various games for negative examples. The learned video marker model is used to detect visual markers from all the video frames in a test game. If a detection is declared for a video frame, a binary number "1" is assigned to this frame. Otherwise "0" is assigned.

We have used the following technique to eliminate some false alarms in video marker detections: for every frame, we look at a range of frames corresponding to 1 second (starting from 14 frames before the current frame to 14 frames after the current frame). If the number of frames that have a

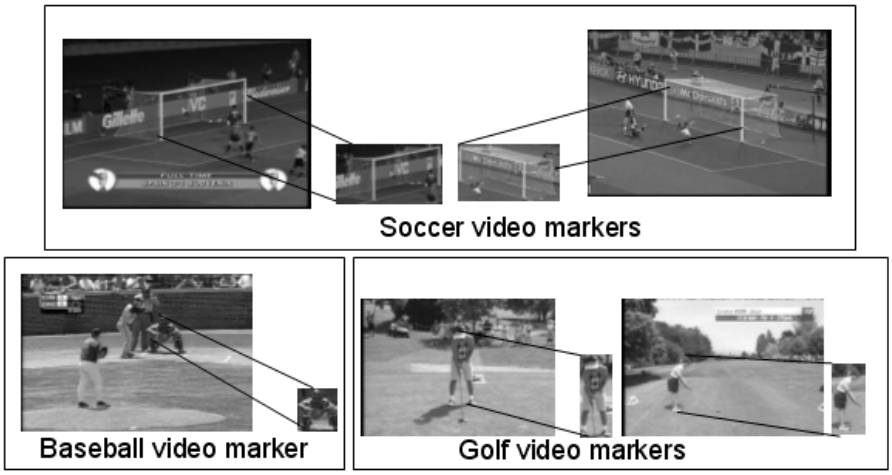

Figure 10: Examples of visual markers for different sports

detection declared, is above a threshold, then we declare this frame as a frame that has detection. Otherwise, we declare this frame as a false positive. By varying this threshold (a percentage of the total number of frames in the range, in this case, 29), we can compare the number of detections with those in the ground truth set (marked by human viewers). We show the precision-recall curve for baseball catcher detection in Fig. 11. We have achieved, for example, around 80% precision for a recall of 70%.

6.2.3 Audio-Visual Markers Negotiation for Highlights Candidates Generation

Ideally each visual marker can be associated with one and only one audio marker and vice versa. Thus they make a pair of audio-visual markers indicating the occurrence of a highlight event in their vicinity. But, since many pairs might be wrongly grouped due to false detections and misses, some post-processing is needed to keep the error to a minimum. We perform the following for associating an audio marker with a video marker.

- If a contiguous sequence of visual markers overlaps with a contiguous sequence of audio markers by a large margin (e.g., the percentage of overlapping is greater than 50%), then we form a "highlight" segment spanning from the beginning of the visual marker sequence to the end of the audio visual marker sequence.
- Otherwise, we associate a visual marker sequence with the nearest audio marker sequence that follows it if the duration between the two is less than a duration threshold (e.g., the average duration of a set of training "highlights" clips from baseball games).

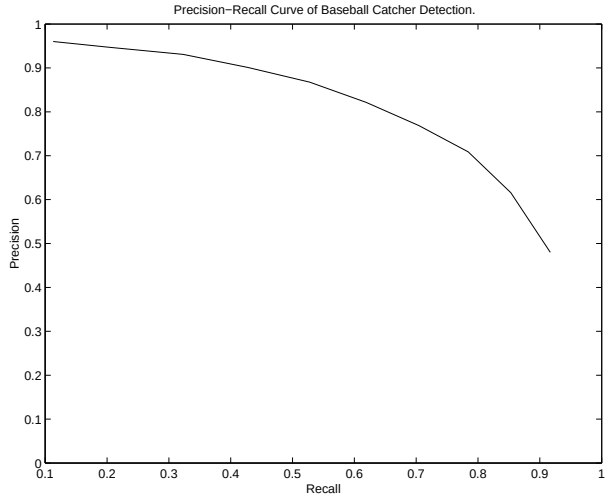

Figure 11: The precision-recall curve of baseball catcher detection

6.2.4 Finer-Resolution Highlights Recognition and Verification

Highlight candidates, delimited by the audio markers and visual markers, are quite diverse. For example, golf swings and putts share the same audio markers (audience applause and cheering) and visual markers (golfers bending to hit the ball). Both of these two kinds of golf highlight events can be found by the aforementioned audio-visual markers detection based method. To support the task of retrieving finer events such as "golf swings only" or "golf putts only", we have developed techniques that model these events using low level audio-visual features. Furthermore, some of these candidates might not be true highlights. We eliminate these false candidates using a finer-level highlight classification method. For example, for golf, we build models for golf swings, golf putts and non-highlights (neither swings nor putts) and use these models for highlights classification (swings or putts) and verification (highlights or non-highlights).

As an example, let us look at finer level highlight classification for a baseball game using low-level color features. The diverse baseball highlight candidates found after the audio markers and visual markers negotiation step are further separated using the techniques described here. For baseball, there are two major categories of highlight candidates, the first being "balls or strikes" in which the batter does not hit the ball, the second being "ball-hits" in which the ball is hit to the field or audience. These two categories have different color patterns. In the first category, the camera is fixed at the pitch scene, so the variance of color distribution over time is low. In the second category, in contrast, the camera first shoots at the pitch scene, then it follows the ball to the field or the audience, so the variance of color distribution over time is higher.

We extract the 16-bin color histogram using the Hue component in the Hue-Saturation-Value (HSV) color space from every video frame of each of the highlight candidate video clip. So every highlight candidate is represented by a matrix of size $L \times 16$ where L is the number of video frames. Let us denote this matrix as the "color histogram matrix". We use the following algorithm to do the finer-resolution highlights classification:

Video Length	540 minutes	
Method	A-V Negotiation	Finer-Resolution Classification
Number of Highlight Candidates	205	173
Number of False Alarms	50	18
Highlights Length	29.5 minutes	25.8 minutes

Table 1: Results before and after the finer-resolution highlights classification step.

- For every color histogram matrix, calculate the "clip-level" mean vector (of length 16) and the "clip-level" standard deviation (STD) vector (also of length 16) over its rows.
- Cluster all the highlight candidate video clips based on their "clip-level" STD vectors into 2 clusters. The cluster algorithm we have used is the k-means algorithm.
- For each of the clusters, calculate the "cluster-level" mean vector (of length 16) and the "cluster-level" STD vector (also of length 16) over the rows of the all the color histogram matrices within the cluster.
- If the value at any color bin of the "clip-level" mean vector is outside the 3σ range of the "cluster-level" mean vector where σ is the STD of the "cluster-level" STD vector at the corresponding color bin, remove it from the highlight candidate list.

When testing on a 3-hour long baseball game, of all the 205 highlight candidates, we have removed 32 using the above algorithm. These 32 clips have been further confirmed to be false alarms by human viewers (see Table 1).

6.3 Video Retrieval

Video retrieval is concerned with how to return similar video clips (or scenes, shots, and frames) to a user given a video query. There are two major categories of existing work. One is to first extract key frames from the video data, then use image retrieval techniques to obtain the video data *indirectly*. Although easy to implement, it has the obvious problem of losing the temporal dimension. The other technique incorporates motion information (sometimes object tracking) into the retrieval process. Although this is a better technique, it requires the computationally expensive task of motion analysis. If object trajectories are to be supported, then this becomes more difficult.

Here we view video retrieval from a different angle. We seek to construct a video Index to suit various users' needs. However, constructing a video Index is far more complex than constructing an index for books. For books, the form of an index is fixed (e.g., key words). For videos, the viewer's interests may cover a wide range. Depending on his or her knowledge and profession, the viewer may be interested in semantic level labels (building, car, people), low level visual features (color, texture, shape), or the camera motion effects (pan, zoom, rotation). In the system described here, we support the following three Index categories:

- Visual Index

- Semantic Index
- Camera motion Index

For scripted content, frame clusters are first constructed to provide indexing to support semantic level and visual feature-based queries. For unscripted content, since audio marker detection and visual marker detection provide information about the content such as whether an image frame has a soccer goal post or whether a segment of audio has the applause sound, we use images or audio segments with audio or video object detection as the visual index.

For scripted content, our clustering algorithm is described as follows:

1. Feature extraction: color and texture features are extracted from each frame. The color feature is an 8×4 2D color histogram in HSV color space. The V component is not used because of its sensitivity to lighting conditions. The H component is quantized finer than the S component due to the psychological observation that the human visual system is more sensitive to Hue than to Saturation. For texture features, the input image is fed into a wavelet filter bank and is then decomposed into de-correlated sub-bands. Each sub-band captures the feature of a given scale and orientation from the original image. Specifically, we decompose an image into three wavelet levels; thus 10 sub-bands. For each sub-band, the standard deviation of the wavelet coefficients is extracted. The 10 standard deviations are used as the texture representation for the image [39].

2. Global clustering: based on the features extracted from each frame, the entire video clip is grouped into clusters. A detailed description of the clustering process can be found in [20]. Note that each cluster can contain frames from multiple shots and each shot can contain multiple clusters. The cluster centroids are used as the visual Index and can be later labeled as a semantic Index (see section 6.3.). This procedure is illustrated in Fig. 12.

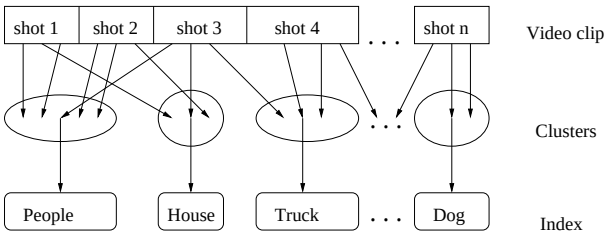

Figure 12: From video clip to cluster to index for scripted content.

After the above clustering process, the entire video clips are grouped into multiple clusters. Since color and texture features are used in the clustering process, all the entries in a given cluster are visually similar. Therefore these clusters naturally provide support for the visual queries.

For unscripted content, we group the video frames based on the key audio/visual marker detectors into clusters such as “frames with a catcher”, “frames with a bat-swing” and “clips with cheering” etc. The video frames or audio clips are then used as visual index or audio index (see Fig. 13).

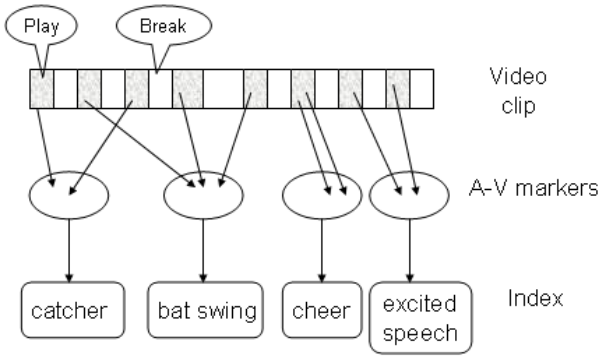

Figure 13: From video clip to cluster to index for unscripted content.

In order to support semantic level queries, semantic labels need to be provided for each cluster. There are two possible approaches. One is based on the Hidden Markov Model (HMM) and the other is an annotation based approach. Since the former approach also needs training samples, both approaches are semi-automatic. To learn details of the first approach, readers are referred to [17]. We will introduce the second approach here. Instead of attempting to attack the unsolved automatic image understanding problem, semi-automatic human assistance is used. We have built interactive tools to display each cluster centroid frame to a human user, who will label that frame. The label will then be *propagated* through the whole cluster. Since only the cluster centroid frame needs labeling, the interactive process is fast. For a 21,717 frame video clip (Movie1), about 20 minutes is needed. After this labeling process, the clusters can support both visual and semantic queries. The specific semantic labels for Movie1 are people, car, dog, tree, grass, road, building, house, etc.

To support camera motion queries, we have developed techniques to detect camera motion in the MPEG compressed domain [40]. The incoming MPEG stream does not need to be fully decompressed. The motion vectors in the bit stream form good estimates of camera motion effects. Hence, panning, zooming, and rotation effects can be efficiently detected [40].

6.4 A Unified Framework for Summarization, Browsing and Retrieval

The above two subsections described video browsing (using ToC generation and highlights extraction) and retrieval techniques separately. In this section, we integrate them into a unified framework to enable a user to go "back and forth" between browsing and retrieval. Going from the Index to the ToC or the Highlights, a user can get the *context* where the indexed entity is located. Going from the ToC or the Highlights to the Index, a user can *pinpoint* specific queries. Fig. 14 illustrates the unified framework.

An essential part of the unified framework is composed of the weighted links. The links can be established between Index entities and scenes, groups, shots, and key frames in the ToC structure for scripted content and between Index entities and finer-resolution highlights, highlight candidates, audio-visual markers and plays/breaks.

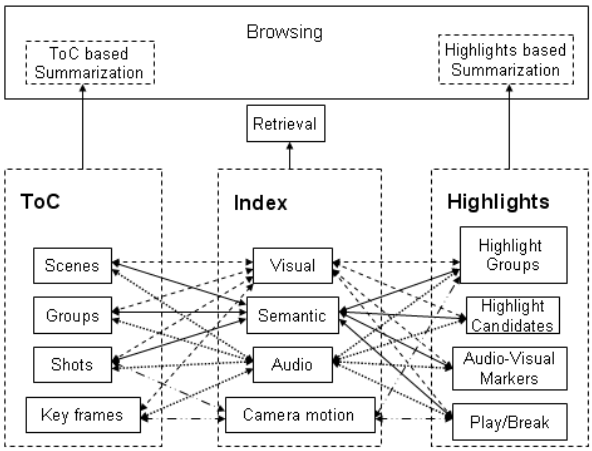

Figure 14: A unified framework

For scripted content, as a first step, in this paper we focus our attention on the links between Index entities and shots. Shots are the building blocks of the ToC. Other links are generalizable from the shot link. To link shots and the *visual Index*, we propose the following techniques. As we mentioned before, a cluster may contain frames from multiple shots. The frames from a particular shot form a sub-cluster. This sub-cluster's centroid is denoted as c_{sub} and the centroid of the whole cluster is denoted as c. This is illustrated in Fig. 15. Here c is a representative of the whole cluster (and thus the visual Index) and c_{sub} is a representative of the frames from a given shot in this cluster. We define the similarity between the cluster centroid and sub-cluster centroid as the link weight between Index entity c and that shot.

$$w_v(i,j) = similarity(c_{sub}, c_j) \tag{7}$$

where i and j are the indices for shots and clusters, respectively, and $w_v(i,j)$ denotes the link weight between shot i and visual Index cluster c_j.

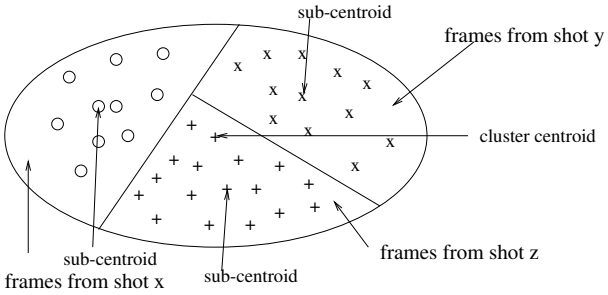

Figure 15: Sub-clusters

After defining the link weights between shots and the visual Index, and labeling each cluster, we can next establish the link weights between shots and the *semantic Index*. Note that multiple clusters may share the same semantic label. The link weight between a shot and a semantic Index

Table 2. From the semantic, visual, camera Index to the ToC.

shot id	0	2	10	12	14	31	33
w_s	0.958	0.963	0.919	0.960	0.957	0.954	0.920
shot id	16	18	20	22	24	26	28
w_v	0.922	0.877	0.920	0.909	0.894	0.901	0.907
shot id	0	1	2	3	4	5	6
w_c	0.74	0.03	0.28	0.17	0.06	0.23	0.09

Table 3. From the ToC (shots) to the Index.

Index	fence	mail box	human hand	mirror	steer wheel
Weight	0.927	0.959	0.918	0.959	0.916

is defined as:

$$w_s(i,k) = max_j(w_v(i,j)) \tag{8}$$

where k is the index for the semantic Index entities; and j represents those clusters sharing the same semantic label k.

The link weight between shots and a *camera motion Index* (e.g., panning) is defined as:

$$w_c(i,l) = \frac{n_i}{N_i} \tag{9}$$

where l is the index for the camera operation Index entities; n_i is the number of frames having that camera motion operation; and N_i is the number of frames in shot i.

For unscripted content, the link weight between plays/breaks and the *visual Index* is defined as:

$$w_{p/b}(i,l) = \frac{m_i}{M_i} \tag{10}$$

where l is the index for the audio/visual Index entities (catcher, goal post, cheering etc.); m_i is the number of frames having that visual index detected and M_i is the number of frames in play/break i.

We have carried out extensive tests using real-world video clips. The video streams are MPEG compressed, with the digitization rate equal to 30 frames/s. Table 2 summarizes example results over the video clip Movie1. The first two rows are an example of going from the semantic Index (e.g., car) to the ToC (shots) The middle two rows are an example of going from the visual Index to the ToC (shots) The last two rows are going from the camera operation Index (panning) to the ToC (shots).

For unscripted content (a baseball game), we show an example of going from the visual Index (e.g., a video frame with a catcher in Fig. 17) to the Highlights (segments with catcher(s)) in Fig. 16. We show another example of going from one highlight segment to the visual Index in Fig. 18. Note that some frames without the catcher have also been chosen because there are audio markers (audience cheering) associated with them.

By just looking at each isolated Index alone, a user usually cannot understand the context. By going from the Index to the ToC or Highlights (as in Table 2 and Fig. 16), a user quickly

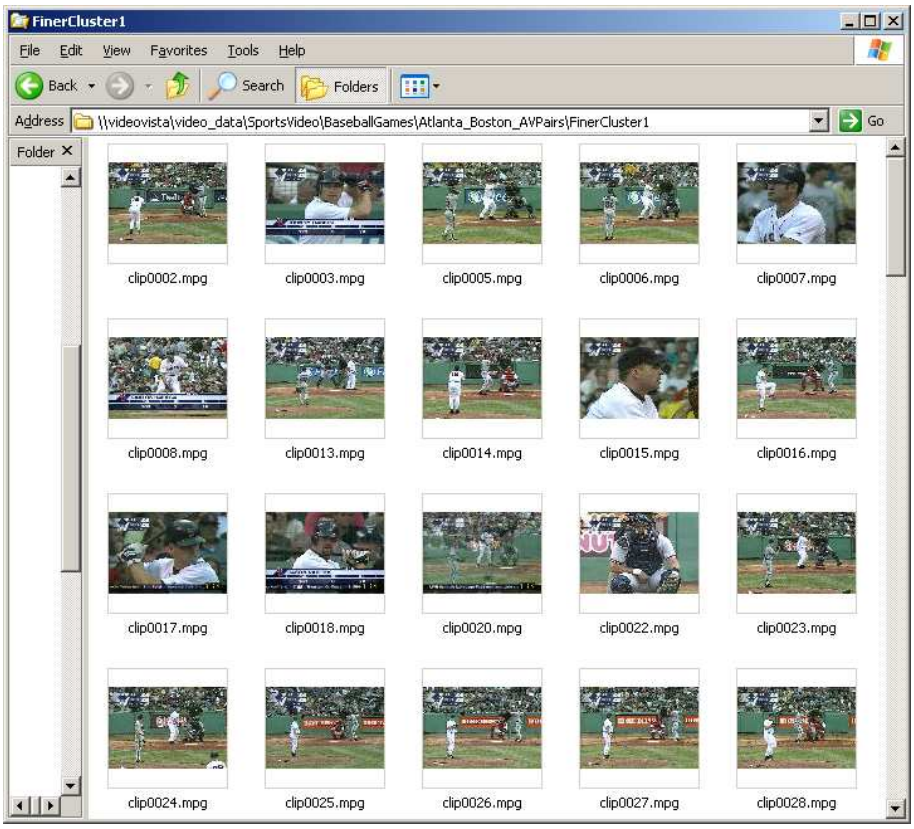

Figure 16: Interface for going from the semantic Index to the Highlights

learns when and under which circumstances (e.g., within a particular scene) that Index entity is happening. Table 2 summarizes how to go from the Index to the ToC to find the *context*. We can also go from the ToC or the Highlights to the Index to *pinpoint* a specific Index. Table 3 summarizes which Index entities appeared in shot 33 of the video clip Movie1.

For a continuous and long medium such as video, a "back and forth" mechanism between summarization and retrieval is crucial. Video library users may have to see the summary of the video first before they know what to retrieve. On the other hand, after retrieving some video objects, the users will be better able to browse the video in the correct direction. We have carried out extensive subjective tests employing users from various disciplines. Their feedback indicates that this unified framework greatly facilitated their access to video content, in home entertainment, sports and educational applications.

7 Conclusions and Promising Research Directions

In this chapter, we:

- Reviewed and discussed recent research progress in multimodal (audio-visual) analysis, rep-

Figure 17: An image used as a visual Index (catcher detected)

resentation, summarization, browsing and retrieval;

- Introduced the video ToC, the Highlights and the Index and presented techniques for constructing them;
- Proposed a unified framework for video summarization, browsing and retrieval; and proposed techniques for establishing the link weights between the ToC, the Highlights and the Index.

We should be aware that video is not just an audio-visual medium. It contains additional text information and is thus "true" multimedia. We need to further extend our investigation to the integration of closed-captioning into our algorithm to enhance the construction of ToCs, Highlights, Indexes and link weights.

8 Acknowledgment

Part of this work (Rui and Huang) was supported in part by ARL Cooperative Agreement No. DAAL01-96-2-0003, and in part by a CSE Fellowship, College of Engineering, UIUC. The authors (Xiong, Radhakrishnan and Divakaran) would like to thank Dr. Mike Jones of MERL for his help in visual marker detection. They also would like to thank Mr. Kohtaro Asai of Mitsubishi Electric Corporation(MELCO), Japan for his support and encouragement and Mr. Isao Otsuka of MELCO, for his valuable application-oriented comments and suggestions. The authors (Rui and Huang) would like to thank Sean X. Zhou, Atulya Velivelli and Roy R. Wang for their contribution.

References

[1] H. Zhang, A. Kankanhalli, and S. W. Smoliar, "Automatic partitioning of full-motion video," *ACM Multimedia Sys.*, vol. 1, no. 1, pp. 1–12, 1993.

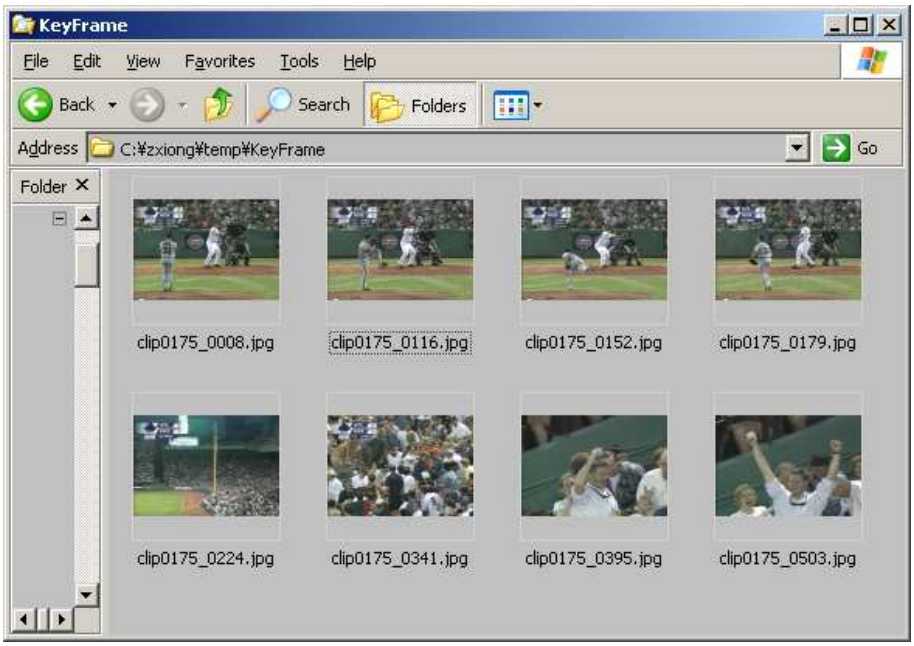

Figure 18: Interface for going from the Highlights to the visual Index

[2] R. M. Bolle, B.-L. Yeo, and M. M. Yeung, "Video query: Beyond the keywords," Technical Report, IBM Research, Oct. 17, 1996.

[3] D. Zhong, H. Zhang, and S.-F. Chang, "Clustering methods for video browsing and annotation," Tech. Rep., Columbia University, 1997.

[4] Y. Rui, T. S. Huang, and S. Mehrotra, "Exploring video structures beyond the shots," in *Proc. of IEEE Conf. Multimedia Computing and Systems*, 1998.

[5] H. Zhang, S. W. Smoliar, and J. J. Wu, "Content-based video browsing tools," in *Proc. IS&T/SPIE Conf. on Multimedia Computing and Networking*, 1995.

[6] L. Xie, S.-F. Chang, A. Divakaran, and H. Sun, "Structure analysis of soccer video with domain knowledge and hidden markov models," *Pattern Recognition Letters*, 2004. to appear.

[7] A. Hampapur, R. Jain, and T. Weymouth, "Digital video segmentation," in *Proc. ACM Conf. on Multimedia*, 1994.

[8] R. Kasturi and R. Jain, "Dynamic vision," in *Proc. of Computer Vision: Principles* (R. Kasturi and R. Jain, eds.), (Washington), IEEE Computer Society Press, 1991.

[9] F. Arman, A. Hsu, and M.-Y. Chiu, "Feature management for large video databases," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1993.

[10] J. Meng, Y. Juan, and S.-F. Chang, "Scene change detection in a mpeg compressed video sequence," in *Proc. SPIE Symposium on Electronic Imaging: Science & Technology- Digital Video Compression: Algorithms and Technologies*, 1995.

[11] B.-L. Yeo, "Efficient processing of compressed images and video," Ph.D. dissertation, Princeton University, 1996.

[12] R. Zabih, J. Miller, and K. Mai, "A feature-based algorithm for detecting and classifying scene breaks," in *Proc. ACM Conf. on Multimedia*, 1995.

[13] A. Nagasaka and Y. Tanaka, "Automatic video indexing and full-video search for object appearances," in *Proc. Visual Database Systems II*, pp. 113–127, Elsevier Science Publishers, 1992.

[14] D. Swanberg, C.-F. Shu, and R. Jain, "Knowledge guided parsing in video databases," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1993.

[15] H. Zhang and S. W. Smoliar, "Developing power tools for video indexing and retrieval," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1994.

[16] H. Zhang, C. Y. Low, S. W. Smoliar, and D. Zhong, "Video parsing, retrieval and browsing: An integrated and content-based solution," in *Proc. ACM Conf. on Multimedia*, 1995.

[17] M. R. Naphade, R. Mehrotra, A. M. Ferman, T. S. Huang, and A. M. Tekalp, "A high performance algorithm for shot boundary detection using multiple cues," in *Proc. IEEE Int. Conf. on Image Proc.*, (Chicago), Oct. 1998.

[18] J. S. Boreczky and L. A. Rowe, "Comparison of video shot boundary detection techniques," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1996.

[19] R. M. Ford, C. Robson, D. Temple, and M. Gerlach, "Metrics for scene change detection in digital video sequences," in *Proc. IEEE Conf. on Multimedia Comput. and Syss*, 1997.

[20] Y. Zhuang, Y. Rui, T. S. Huang, and S. Mehrotra, "Adaptive key frame extraction using unsupervised clustering," in *Proc. IEEE Int. Conf. on Image Proc.*, 1998.

[21] P. O. Gresle and T. S. Huang, "Gisting of video documents: A key frames selection algorithm using relative activity measure," in *Proc. the 2nd Int. Conf. on Visual Information Systems*, 1997.

[22] W. Wolf, "Key frame selection by motion analysis," in *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc.*, 1996.

[23] A. Divakaran, K. Peker, R. Radhakrishnan, Z. Xiong, and R. Cabasson, "Video summarization using MPEG-7 motion activity and audio descriptors," *Video Mining*, January 2003. eds. A. Rosenfeld, D. Doermann and D. DeMenthon, Kluwer Academic Publishers.

[24] L. Xie, S. Chang, A. Divakaran, and H. Sun, "Structure analysis of soccer video with hidden markov models," *Proc. Intl. Conf. on Acoustic, Speech and Signal Processing, (ICASSP-2002)*, May 2002. Orlando, FL, USA.

[25] T. C.Wren, A.Azarbayejani and A.Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. vol 19, pp. 780–785, July 1997.

[26] Z. Xiong, R. Radhakrishnan, and A. Divakaran, "Effective and efficient sports highlights extraction using the minimum description length criterion in selecting gmm structures," *accepted into Intl' Conf. on Multimedia and Expo(ICME)*, June 2004.

[27] Y. Rui, A. Gupta, and A. Acero, "Automatically extracting highlights for TV baseball programs," *Eighth ACM International Conference on Multimedia*, pp. 105 – 115, 2000.

[28] S. Nepal, U. Srinivasan, and G. Reynolds, "Automatic detection of 'goal' segments in basketball videos," in *Proceedings of the ACM Conf. on Multimedia*, 2001.

[29] Y.-L. Chang, W. Zeng, I. Kamel, and R. Alonso, "Integrated image and speech analysis for content-based video indexing," in *Proceedings of the IEEE Intl' Conf. Multimedia Computing and Systems*, June 1996.

[30] T. I. T. Kawashima, K. Tateyama and Y. Aoki, "Indexing of baseball telecast for content-based video retrieval," in *Proc. IEEE Int. Conf. on Image Proc.*, 1998.

[31] Y. Gong, L. Sin, C. Chuan, H. Zhang, and M. Sakauchi, "Automatic parsing of TV soccer programs," *IEEE International Conference on Multimedia Computing and Systems*, pp. 167 – 174, 1995.

[32] M. Yeung, B.-L. Yeo, and B. Liu, "Extracting story units from long programs for video browsing and navigation," in *Proc. IEEE Conf. on Multimedia Comput. and Syss*, 1996.

[33] M. Irani and P. Anandan, "Video indexing based on mosaic representations," *Proceedings of IEEE*, vol. 86, pp. 905–921, May 1998.

[34] Y. Rui, T. S. Huang, and S. Mehrotra, "Constructing table-of-content for videos," *Journal of Multimedia Sys.*, vol. 7, pp. 359–368, Sept 1999.

[35] C.-W. N. A. Velivelli and T. S. Huang, "Detection of documentarty scene changes using audio-visual fusion," in *International Conf on Image and Video Retrieval.*, (Urbana), July. 2003.

[36] H. Sundaram and S. Chang, "Audio scene segmentation using multiple models, features and time scales," *In IEEE International Conference on Acoustics, Speech, and Signal Processing*, June 2000. Istanbul, Turkey.

[37] C.-W. Ngo, "Analysis of spatio-temporal slices for video content representation,," Ph.D. dissertation, Hong Kong university of science and technology, 2000.

[38] P. Viola and M. Jones, "Robust real-time object detection," *Second International Workshop on Statistical and Computational Theories of Vision - Modeling, Learning, Computing and Sampling*, July 2001. Vancouver, Canada.

[39] Y. Rui, T. S. Huang, and S. Mehrotra, "Content-based image retrieval with relevance feedback in MARS," in *Proc. IEEE Int. Conf. on Image Proc.*, 1997.

[40] J. A. Schmidt, "Object and camera parameter estimation using mpeg motion vectors," M.S. thesis, University of Illinois at Urbana-Champaign, 1998.