Columbia University ADVENT Technical Report #091, Dec. 2000.

Structure Parsing and Event Detection for Sports Video

Di Zhong and Shih-Fu Chang

Department of Electrical Engineering, Columbia University

December 27, 2000

1 Introduction

For video indexing, typically videos are first processed to obtain constituent objects and features. These extracted entities provide an intermediate content model to effectively describe videos. In earlier chapters, we have studied temporal segmentation that generates elementary video shots, and spatial-temporal segmentation that extracts video regions and objects. We also developed feature matching algorithms and query methods to support visual similarity search. These works provide very useful tools for accessing online digital videos.

However, the objects and features we have extracted so far contain little information at the semantic level. To solve this problem, we can explore the knowledge and constraints in specific domain and apply domain-specific rules and/or unsupervised machine learning techniques. In this chapter, we present an event detection and structure parsing system for sports videos. This system is built on top of segmentation and search techniques we have presented in prior chapters. We also demonstrate a summarization and browsing interface that allows users to easily access content structure and event index of video data.

Combining domain knowledge and automatically extracted low-level features have been seen in many works. In [7], an edge model is used to match anchorperson views in specific news programs. The story structure is then reconstructued by finding anchorperson views as well as commercials.

In [2], logical story units (LSU's) are extracted from movies according to global temporal consistency. The consistency is based on the assumption that an event is related to a specific location and to certain characters. Sports videos have different characteristics. A sports game usually occurs in one specific playground, contains abundant motion information and has well-defined content structures. Event detection in basketball and tennis videos has been studied in [5] and [6] respectively, but without attempts to reconstructing high-level structures. We will further discuss these two in later sections.

In this chapter, we present a real-time structure parsing and event detection system for sports videos. Compared to existing work, our solution has the following unique features.

- A general framework for video structure discovery and event detection.
- Combination of domain-specific knowledge and generic machine learning techniques.
- Multi-stage scene detection algorithms using our unique moving object segmentation and feature extraction methods.
- Real time processing performance by exploring redundancy in the compressed video streams.
- Higher accuracy demonstrated in specific sports domains such as tennis and baseball.

We will present our real-time prototype systems demonstrating results in tennis and baseball videos.

2 Semantic Content in Sports Video

Sports video is a major part in most broadcasting TV programs, and have a large number of audience. Compared to other videos such as news and movies, sports videos have well-defined content structure and domain rules.

First, sports videos are structured. A long sports game is often divided into a few segments. Each segment may again contain some sub-segments. Furthermore, there are a fixed number of cameras in the field which result in unique scenes during each segment. For example, in football, a game contains two halves, and each half has two quarters. Within each quarter, there are many plays, and each play starts with the formation in which players line up on two sides of the ball. A tennis game is divided first into sets, then games and serves (**Figure 1**). Usually when a serve starts, the scene is switched to court views (**Figure 2**). In addition, for TV broadcastings, there are commercials or other special information (e.g., score board, player's name) inserted between segments.

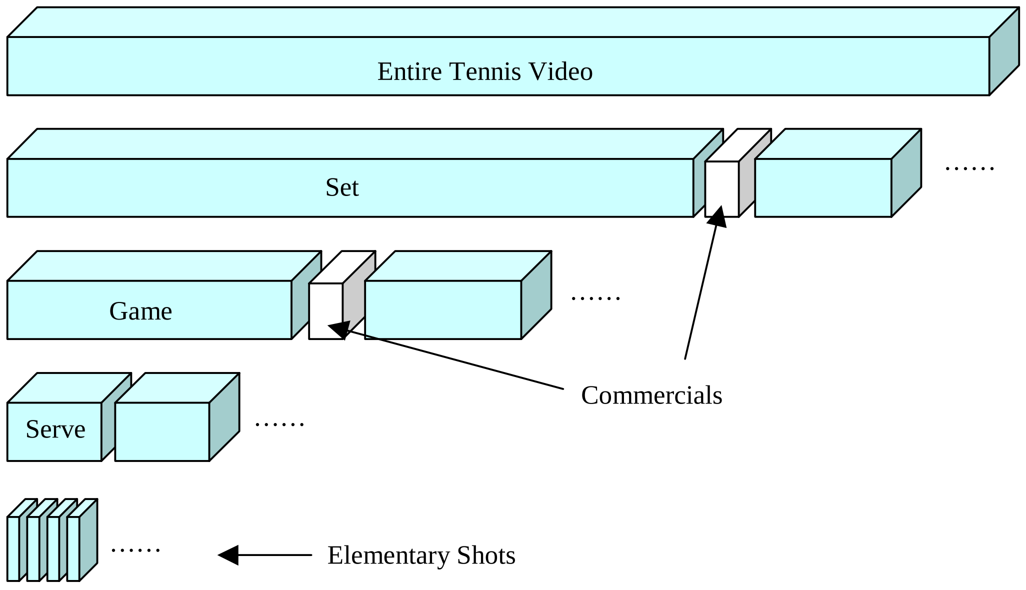


Figure 1 The temporal structure of a typical tennis video

The above characteristics of specific domains provide some opportunities for developing new algorithms and tools to parse video structures, and to reconstruct some kinds of build table-of-contents for sports video inedxing. As the structure is fixed given a type of sports, domain models can be used to greatly improve detection accuracy. The resulting systems and tools will enable users to have access any arbitary segment at the desired level based on the intuitive and informative structure.

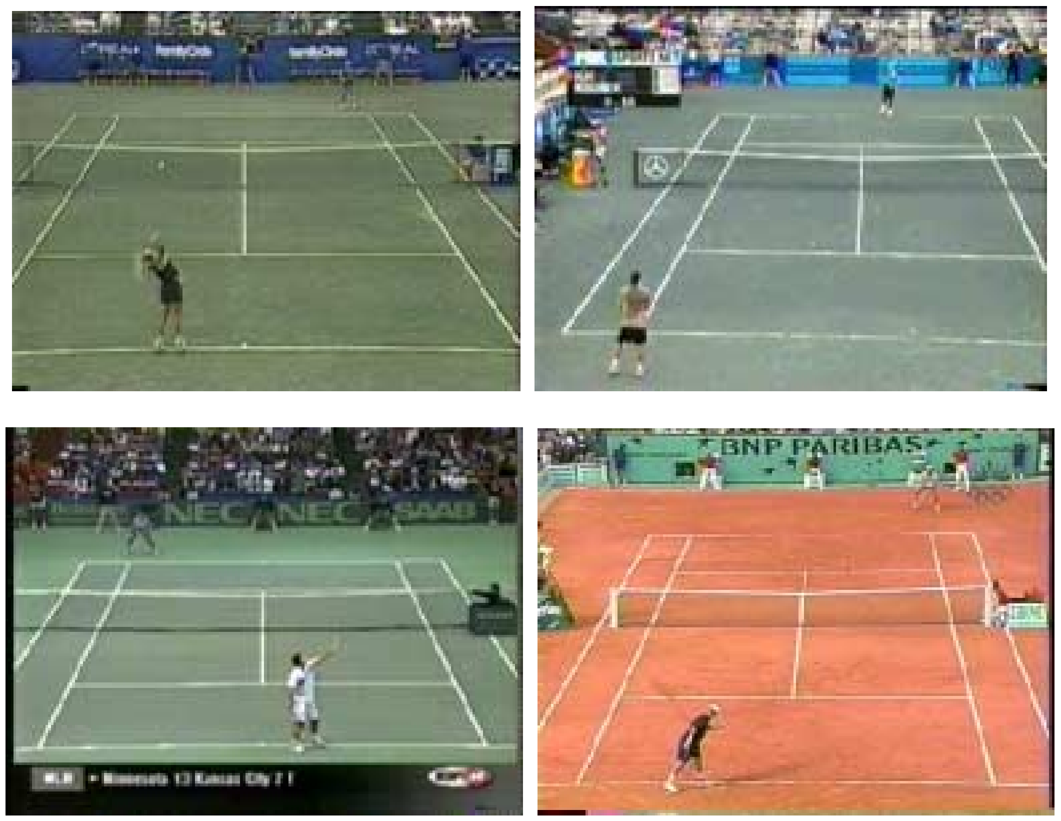


Figure 2 Scenes from four different tennis games

Another property of sports videos is that there are many special events in the video that are interesting to most audience. Typical examples include touch-downs in football, shots in basketball, home-runs in baseball, aces in tennis and so on. Within one sports, an event usually has similar visual-audio attributes that do not vary greatly from game to game. Baseball games are played in grass fields, and when there is a home-run, the camera will quickly follow the ball across the field and there will be loud screams from the audience. Tennis courts typically have uniform ground colors and white court lines, and within a serve, the camera is typically placed at shots from the top-rear position (e.g., a few scenes from different games are shown in **Figure 2**). These characteristics allow us to adopt some domain-specific rules and apply machine-learning techniques using automatically segmented objects and features to detect interesting events. Hence, given a sports video, we can identify some important events that are interesting to most audience, and automatically or semi-automatically build an index to all the occurrences of these events. Similar to the keyword index in a book, this event index enables viewers to access desired scenes much more intuitively and efficiently compared to traditional fast forward and backward search.

A few works have been conducted to analyze sports videos. In [5], a basketball annotation system is presented. Using prior knowledge of basketball games, the system detects wide-angle shots versus close-up shots. Camera motions are further analyzed within wide-angle shots and used to detect possible fast break, steal and probable shots. When users want to retrieve one type of event (e.g., fast break), only sequences satisfying corresponding motion criteria are shown. Tennis videos are studied in [6]. The approach is based on the extraction of a model for the tennis court-lines from videos. A player tracking algorithms is developed to track players, and then a reasoning module is used to map low-level positional information to high-level tennis play events.

The issue addressed in above researches is to detect certain events using low-level features such as motion and position. The well-defined temporal structures were not explored. Another challenging problem that has not been well addressed is the real time constraint. Sports video broadcasting is mostly live, and interests of audience go down greatly after the results are known. The real-time capability in generating event indices is critical in practical applications for sports videos.

In this chapter, we present a general framework to parse temporal structure of live broadcasted sports videos. In the proposed approach, parsing rules are first built through a training phase. In the operation phase, these rules are updated to adapt to new data in live inputs. We will also show innovative event detection tools using segmentation techniques we have developed in previous chapters.

3 Real Time Video Analysis System

We briefly describe the system architecture of the real time sports video analysis platform. Details about how segmentation and feature extraction algorithms are adapted to meet the real-time requirement will be included in later sections where structure parsing and event detection are discussed.

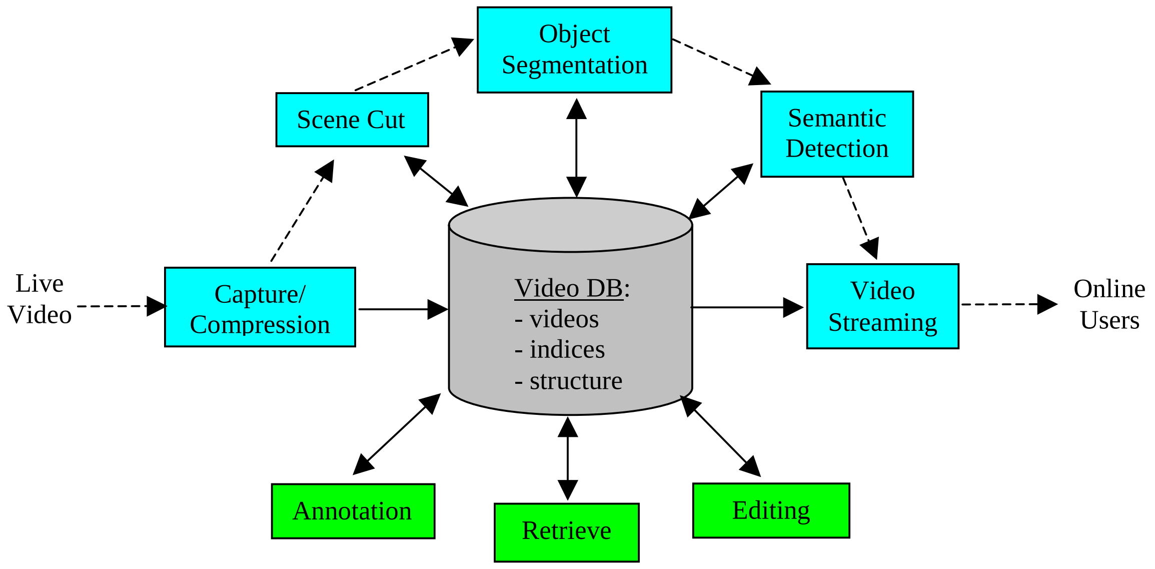


Figure 3 System architecture of the real time video analysis platform

The system is built on a personal computer with Microsoft Window NT 4.0. Real-time performance is reached on a Pentium-III 600 dual CPU machine. The system allows users to build digital video database taking regular analog video sources such as TV broadcast, cable, VCR, or video camera. As shown in **Figure 3**, the dashed arrows indicate real-time flow of video data. Some main function modules are as follows.

- Capture and Compression – An MPEG compression card is used to capture and compress video and audio signals. Text decoder card can be added to obtain closed caption signal.

- Scene Cut Detection – The scene cut detection scheme presented in chapter 2 is used. It detects scene cuts first in MPEG compressed domain with partial decoding. For gradual scene changes, chrominance and edge features are compared in the un-compressed domain. To achieve real-time performance, we only decompress and analyze I-frames. The output scene cut positions and representative frames (e.g., the first I-frame after a scene cut) are stored in the video database.
- Object Segmentation and Feature Extraction – Object segmentation and feature extraction are the most time-consuming processes, and cannot be applied to all frames. To spot important scenes, representative frames are always examined first. When a potential candidate is located, segmentations and more detailed features are computed and used in the subsequent semantic detection module. Thus, although the mission of this module is to extract visual features, it requires a semantic filtering component. The component depends on domain knowledge and should use only simple features that can be quickly computed, such as global color histograms.
- Semantic Detection – The detection tasks can be classified into two categories: one is to parse the structure information; another is to find interesting events. The detection process can also be divided into 1) local, which is applied to individual frames; 2) temporal, which is applied to one or more videos shots. The former one usually precedes the later one. All detected results are stored into the video database to construct structure table or event index.
- Video Streaming –Videos are delivered to users in real-time together with structure parsing and event detection results. The semantic and event information can be used to filter the video content and send a personalized version of the video to each individual user.
- Annotation, Editing – These are usually post processing with manual interactions. However, they may be applied in real time too. For example, when certain events or objects are detected,

additional information such as hyperlinks or highlights can be immediately inserted or overlaid to the original streams. It is also possible to cut and/or replace some portions of broadcasted videos in real time (as in advertisement insertion).

- Retrieve – With our real-time analysis system, a live video can be searched right after, or even during the broadcasting time.

In summary, our scene analysis algorithms work as modules along a streaming pipeline. Such a stream-line architecture is important in applications that may need real-time performance or application specific customization.

The above real-time parsing and indexing system has a lot of applications. It can be used to create large searchable video archives. It can also be used in web-casting of sports programs in which game outlines and scene highlights can be rapidly developed on web sites. The system can also be combined with existing sports annotation software, which allow reporters to annotate and log a game in real time. In interactive applications, interactive user interface with virtual-reality like environments can be enhanced with simultaneous displays of graphics and synchronized video clips.

4 Structure Parsing

Given the structure model of a game, we need to detect visual cues that indicate the beginning and/or ending of each section or sub-section. Some common features occurring between top-level sections are commercials, embedded texts and special logos. Many methods have been proposed for commercial and text detection [3,4]. Here we will study the detection of basic units within a game, such as serves in tennis. These units usually start with a special scene. Simple color based approaches have been suggested [6]. Based on our experiments to be described later, this type of approaches can only reach about 80 percent accuracy. Furthermore, as color information varies from game to game, adaptive methods need to be exploited to handle such variation. In our real-time framework, we first use a fast adaptive color filtering method to select possible candidates, then apply more complicated features to verify them against domain specific rules or models. In the following we present the system using serves in tennis as an example.

4.1 Color Based Adaptive Filtering

Color based filtering is applied to key frames of video shots. First, the filtering models are built through a clustering based training process. The training data should contain enough games so that a new game will be similar to some in the training set. Assume h_i, $i=1\ldots,N$ are color histograms of all serve scenes in the training set. A k-means clustering is used to generate K models (i.e., clusters), $M_1,\ldots,M_K$, such that,

$$h_i \in M_j \ , \quad if \;\; D(h_i, M_j) = \min_{k=1}^{K}(D(h_i, M_k)) \tag{1}$$

where $D(h_i, M_k)$ is the distance between h_i and the mean vector of M_k, i.e. $H_k = \frac{1}{|M_k|}\sum_{h_i \in M_k} h_i$ and $|M_k|$ is the number of training scenes being classified into the model M_k. This means that for each module M_k, H_k is used as its the representative feature vector.

When a new game coming in, proper models need to be chosen to spot serve scenes. This raises a typical egg-and-chicken problem, as we need to know serve scenes to select a correct model. To solve the problem, we detect the first L serve scenes using all models, $M_1,\ldots,M_K$. That is to say, all models are used in the filtering process. If one scene is close enough to any model, the scene will be passed through to subsequent verification processes (Eq 2).

$$h_i^{'} \in M_j, \quad if \ D(h_i^{'}, M_j) = \min_{k=1}^{K}(D(h_i^{'}, M_k)) \ \ and \ \ D(h_i^{'}, M_j) < TH \tag{2}$$

where $h_i^{'}$ is the color histogram of the *i*th shot in the new video, and *TH* is a given filtering threshold to accept shots with enough color similarity. Shot *i* is detected as a serve scene if the segmentation based verification, which will be described in 4.2, is also successful, and we mark this serve as being founded by model M_j (i.e., classify the scene into the model M_j). If the verification fails, $h_i^{'}$ is removed from the set of M_j.

After *L* serve scenes are detected, we find the model M_o, which leads to the search for the model with the most serve scenes.

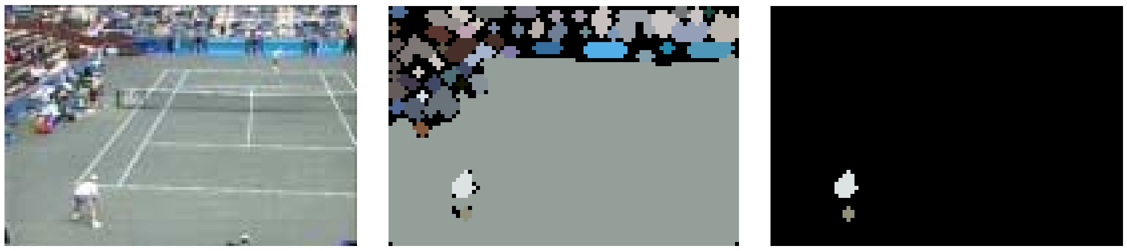
$$|M_o| = \max_{k=1}^{K}(|M_k|) \tag{3}$$

where $|M_k|$ is the number of incoming scenes being classified into the model M_k. In the filtering process for subsequent shots, model M_o and a few of its most close neighboring models are applied in a same way as define in Eq 2.

4.2 Segmentation Based Verification

The salient feature region extraction and moving object detection we proposed in chapter 3 and section 4.3 can be utilized here to produce localized spatial-temporal features. Compared with global features, such as color histograms, spatial-temporal features are more reliable and invariant to detect given scene models. Especially in sports videos, special scenes are often made of several objects at fixed locations. The similarity matching scheme of visual and structure features we studied in previous chapters can also be easily adapted here for model verification.

To verify serve scenes, the moving object detection algorithm (section 4.3) is applied. To achieve real-time performance, segmentation is performed on the down-sampled images of the key frame (which is chosen to be an I-frame) and its successive P-frame. The down-sampling rate used in our experiment is 4, both horizontally and vertically, which results in images with size 88x60. Motion fields are estimated using the hierarchical approach as proposed in [1]. An example of segmentation and detection results is shown in **Figure 4**.



a. original key frame b. region segmentation c. moving object detection

Figure 4 An example of automatic region segmentation and moving object detection

Figure 4 (b) shows the region segmentation result. The court is segmented out as one large region, while the player closer to the camera is also extracted. The court lines are not preserved due to the down-sampling. Black areas shown in (b) are tiny regions being dropped at the end of segmentation process. Figure 4 (c) shows the moving object detection result. In this example, the result is quite satisfactory, and only the desired player is detected. Sometimes a few background regions may also be detected as foreground moving object. We will address this issue in section 5 when discussing the player tracking algorithm. Here for verification purpose, as we will describe below, the important thing is not to miss the player.

Following rules are applied in scene verification. First, there must be a large region (e.g. larger than two-thirds of the frame size) with consistent color (or intensity for simplicity). This large region corresponds to the tennis court. The uniformity of a region is measured by the intensity variance of all pixels within the region (Eq 4).

$$Var(p) = \frac{1}{N}\sum_{i=1}^{N}[I(p_i) - \bar{I}(p)]^2 \qquad (4)$$

where N is the number of pixels within a region *p*. $I(p_i)$ is the intensity of pixel *I and* $\bar{I}(p)$ is the average intensity of region *p*. If *Var(p)* is less than a given threshold, the size of region *p* is examined to decide if it corresponds to the tennis court.

Secondly, the size and position of player are examined. The condition is satisfied if a moving object with proper size is detected within the lower half part of the previously detected large “court” region. In a downsized 88x60 image, the size of a player is usually between 50 to 200 pixels. As our detection method is applied at the beginning of a serve, and players are always at the bottom line to start a server, the position of a detected player has to be within the lower half part of the court.

4.3 Edge Based Verification

One unique characteristic of serving scenes in tennis game is that there are horizontal and vertical court lines. Ideally if a camera shots straightforward from top-rear point of the court and all court lines are captured (**Figure 1**), rules for a complete court can be used to verify serve scenes with high precision. However, in a real scene, due to the camera panning and zooming, or object occlusion, usually not all court lines are viewable. Trying to match a full court will result in a low recall rate of serve scenes.

Since we already go through color based filtering and region based verification processes, relatively loosen constraints are enforced on court lines. An example of edge detection using the 5x5 Sobel operator is given in **Figure 5**.

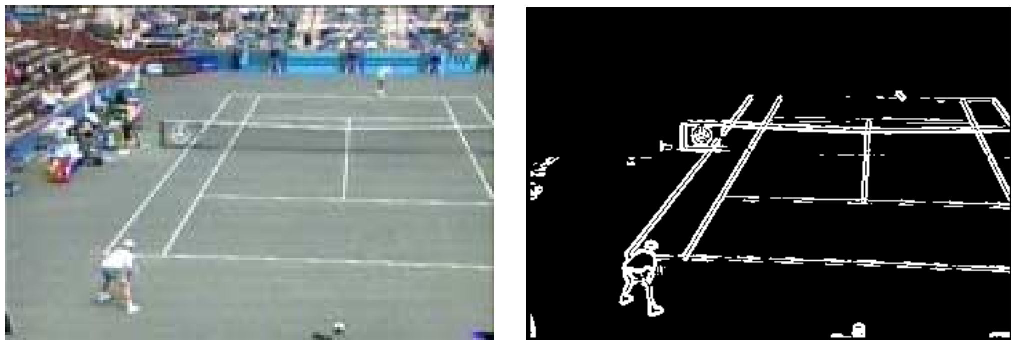


Figure 5 Edge detection within the court region

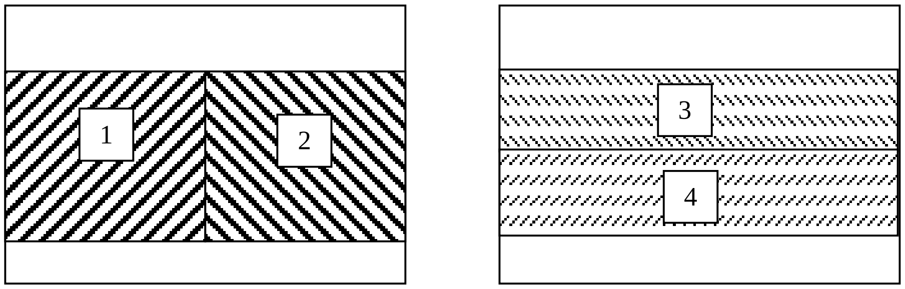


Figure 6 Local windows for hough-transform based line detection

Note that the edge detection is performed on a down-sampled (usually by 2) image and inside the detected court region. Hough transforms are conducted in four local windows to detection straight lines (**Figure 6**). Windows 1 and 2 are used to detect vertical court lines, while windows 3 and 4 are used to detect horizontal lines. It greatly increases the accuracy in detecting straight lines to use local windows instead of a whole frame. As shown in the figure, each pair of windows roughly cover a little bit more than half of a frame, and are positioned somewhat close to the bottom border. This is based on the observation of the usual position of court lines within court views.

The verifying condition is that there are at least two vertical court lines and two horizontal court lines being detected. Note these lines have to be apart from each other for a certain distance, as noises and errors in edge detection and Hough transform may produce duplicated lines. This is based on the assumption that despite of camera panning, there is at least one side of the court, which has two vertical lines, being captured in the video. On the other hand, camera zooming will always keep two of three horizontal lines, i.e., the bottom line, middle court line and net line, in the view.

4.4 Experiments and Discussion

The above scheme shows very good performance on tennis videos. Our experiment on a one-hour tennis video is shown in Table 1. The recall rate is about 93% and the precision is about 97%.

Table 1 Detection result for serve scenes

# of Serve Scene (ground truth)	# of Miss	# of False
89	7	2

This result is very good compared to approaches using only colors. Based on our experiments, previously proposed approaches using color histogram filtering can only achieve about 80% precision rate in order to obtain near 100% recall. No serve scenes should be dropped in the first filtering stage (i.e., recall rate is maximized), thus a low filtering threshold is used. Furthermore, despite of using advanced segmentation and feature extraction, our proposed scene detection and verification process is performed in real time.

One thing to point out here is that the color models used in the first filtering stage include clusters extracted from the first 10 minutes of the testing video as well as some videos from three other games (from different channels or games). As different games may have different color distributions, in real applications, we need to have a fairly comprehensive color models extracted from different games in order to properly process new incoming games. While this is a critical issue for color only approaches, our approach is quite flexible in handling such situations. Considering there are only limited types of play fields, we can lower our filtering threshold to allow more false alarms in the first stage, and rely on the feature based verification process to improve precision rate.

5 Event Detection

In parsing the game structures, we have presented a scene detection approach using spatial-temporal features. In this section, focus will be on detecting and summarizing what happened in an interesting scene. Especially, we will adapt our moving object detection algorithm to track a tennis player in real-time, and analyze his or her trajectory to obtain certain play facts.

5.1 Player Tracking

In chapter 4, we presented an automatic moving object detection method that contains two stages: an iterative motion layer detection step being performed at individual frames; and a temporal detection process combining multiple local results within an entire shot. Here we adapt this approach to track tennis players within court view in real time. Currently we focus on the player who is close to the camera. The player at the opposite side is smaller and not always in the view. It is harder to track small regions in real time because we need to down-sample spatially to reduce computation complexity.

The local motion layer detection process is similar to what we previously explained in 4.2 (**Figure 5**), down-sampled I- and P-frames are segmented and compared to extract motion layers. The reason to skip B-frames is because bi-direction predicted frames require more computation to decode. To ensure real-time performance, only one pair of anchor frames are processed every half second. For a MPEG stream with a GOP size of 15 frames, the I-frame and its immediate following P-frame are used. Motion layer detection is not performed in later P frames in the GOP. This change requires a different temporal detection process to detection moving objects. The process is described as follows.

As half second is a rather large gap for the estimation of motion fields, motion-based region projection and tracking from I frame to another I frame are not reliable, especially when a scene contains fast motion. Thus, a different process is required to match moving layers detected at individual I-frames. We use the following temporal filtering process to select and match objects that are detected at I frames.

Assume O_i^k is the *kth* object (*k=1,…,K)* at the *ith* I-frame in a video shot, $\vec{p}_i^k$, $\vec{c}_i^k$ and s_i^k are the center position, mean color and size of the object respectively. We define the distance between O_i^k and another object at *jth* I-frame, O_j^l, as weighted sum of spatial, color and size differences.

$$D(O_i^k, O_j^l) = w_p \left\| \vec{p}_i^k - \vec{p}_j^l \right\| + w_c \left\| \vec{c}_i^k - \vec{c}_j^l \right\| + w_s \left| s_i^k - s_j^l \right| \tag{5}$$

where w_p, w_c and w_s are weights on spatial, color and size differences respectively. If $D(O_i^k, O_j^l)$ is smaller than a given threshold, *O_TH*, objects O_i^k and O_j^l match with each other. We then define the match between an object with its neighboring I-frame $i+\delta$ as follows,

$$F(O_i^k, i+\delta) = \begin{cases} 1 & \exists\, O_{i+\delta}^l,\ D(O_i^k, O_{i+\delta}^l) < O\_TH \\ 0 & otherwise \end{cases} \tag{6}$$

where $\delta = \pm 1, ..., n$. Let $M_i^k = \sum_{\delta=\pm 1,...,n} F(O_i^k, i+\delta)$ be the total number of frames that have matches of object O_i^k (*k=1,…,K)* within the period $i-\delta$ to $i+\delta$, we select the object with maximum M_i^k. This means that if $M_i^r = \max_{k=1,...,K}(M_i^k)$, the *rth* object is kept at the *ith* I-frame. The other objects are dropped. The above process can be considered as a general temporal median filtering operation.

After the above selection, we obtain the trajectory of the lower player by sequentially taking the center coordinates of the selected moving objects at all I-frames. There are some issues that need some discussion here. First, if no object is found in a frame, linear interpolation is used to fill the missing point. When there are more than one objects being selected at a frame (in the situation when more than one objects have the same maximum number), the one that is spatially close to its precedent is used. In addition, for speed reason, instead of using affine model to compensate camera motion, here we use the detected net lines to roughly align different instances.

Experimental tracking results of one sever scene is shown in **Figure 7**.

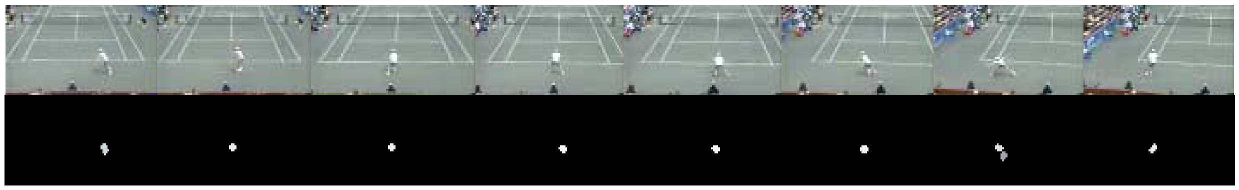


Figure 7 Tennis player tracking within a serve scene (results are shown on 8 I-frames with a 15-frame interval)

The first row shows down sampled frames. The second row contains final player tracking results. The body of the player is well tracked and detected. Successful tracking of tennis players provides a foundation for high-level semantic analysis. Compared with the tracking algorithm in [6], which computes residual errors to find moving objects and then searches players in pre-defined windows, our approach provides more accurate as well as real time performance.

5.2 Trajectory Analysis

The extracted trajectory is analyzed to obtain play information. Presently, we focus on two aspects. The first one is the position of a player. As players usually play at bottom lines, we want to find cases when a player moves to the net zone. The second one is to estimate the number of strikes a player conducted within a serve. Users who want to learn stroke skills or play strategies may be interested in serves with more strikes.

Given a trajectory containing K coordinates, $\vec{p}_k$ $(k=1,\ldots,K)$, at K successive I-frames, we first detect “still points” and “turning points”. $\vec{p}_k$ is a still point if,

$$\min(\|\vec{p}_k - \vec{p}_{k-1}\|, \|\vec{p}_k - \vec{p}_{k+1}\|) < TH \tag{7}$$

where *TH* is a pre-defined threshold. Furthermore, two consecutive still points are merged into one. If point $\vec{p}_k$ is not a still point, the angle at the point is examined. $\vec{p}_k$ is a turning point if

$$\angle(p_k p_{k-1}, p_k p_{k+1}) < 90^o \tag{8}$$

An example of object trajectory is shown in **Figure 8**. After detecting still and turning points, we use them to judge the player’s positions. If there is a position close to the net line (vertically), the serve is classified as net-zone play. The estimated number of strokes is the sum of the numbers of turning and still points.

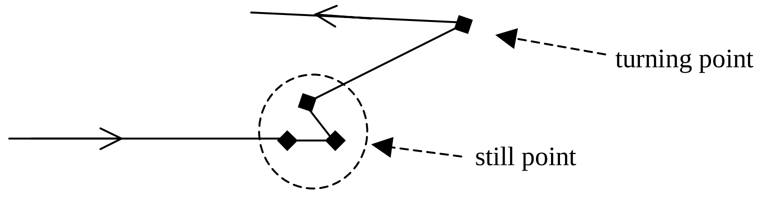


Figure 8 Detection of still and turning points in object trajectory

Experiment results of the above one-hour video are given in Table 2.

Table 2 Trajectory analysis results for one hour tennis video

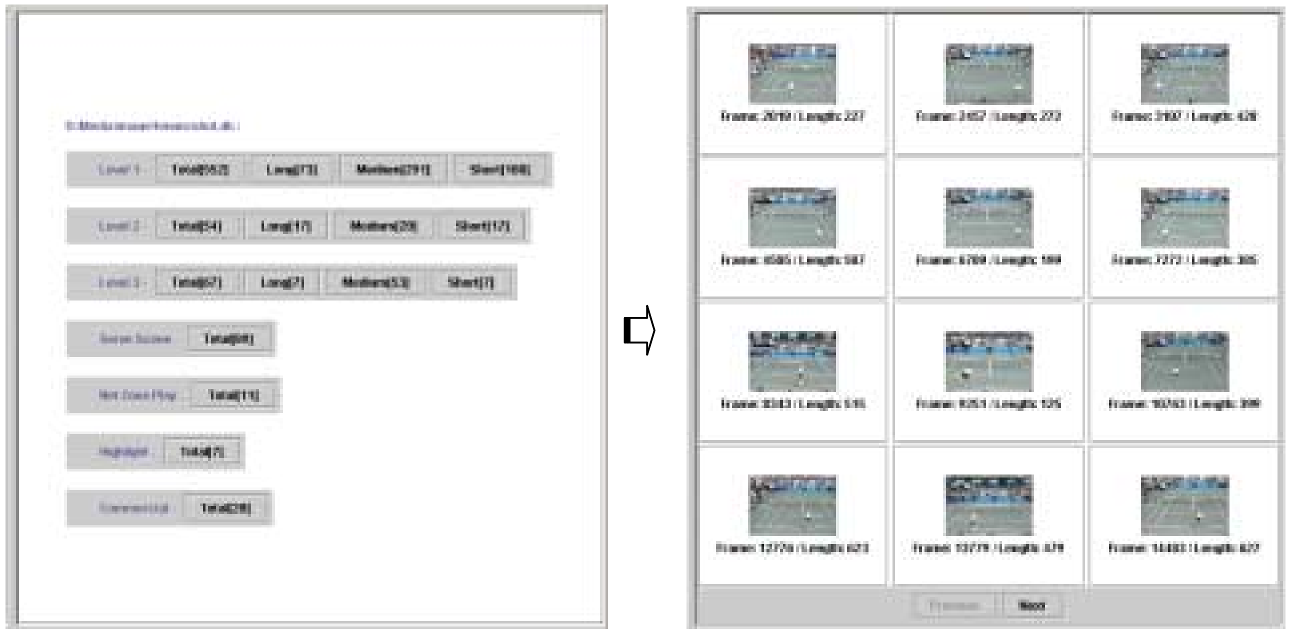
	# of Net Plays	# of Strokes
Ground Truth	12	221
Correct Detection	11	216
False Detection	7	81

In the video, the ground truth includes 12 serves with net play within about 90 serve scenes (see Table 1), and totally 221 strokes in all serves. Most net plays are correctly detected. False detection of net plays is mainly caused by incorrect extraction of player trajectories or court lines. Stroke detection has a precision rate about 72%. Beside the reason of incorrect player tracking, some errors are caused by limitations of our estimation model. First, at the end of a serve, a player may or may not strike the ball in his or her last movement. Many serve scenes also show players walking in the field after the play. In addition, a sever scene sometimes contain two serves if the first serve failed. These may cause problems since currently we detect strokes based on the movement information of the player. To solve these issues, more detailed analysis of motion such as speed, direction, repeating patterns in combination with audio analysis (e.g., hitting sound) will be very useful.

6 A Summarization and Browsing Interface

As we have observed, video data contain large amount of visual and semantic information. Even after content indices are generated, how to show these indices in a limited display is still a challenging issue. In this section, we present a system for video browsing and summarization using the structure parsing and event detection results presented in this thesis. The system has two unique access methods for users to find desired video shots.

First, the system provides a summarization interface (**Figure 9a**). It shows the statistics of video shots, such as number of long, intermediate and short shots. It also shows the number of some common kinds of scenes in a specific domain. For instance, in tennis, there are serve, net-zone play, commercial and etc.). Seeing these summaries, users may follow up with more specific request by choosing a category (e.g., serve). As shown in **Figure 9b**, from here users can directly go to any interesting scenes.



(a) (b)

Figure 9 Summarization interface providing scene index to video

The second interface combines the sequential temporal order and the hierarchical structure among all video shots within a video. As shown in **Figure 10**, the structure tree is in the left window. In this example, games are listed at the top level. There are commercial breaks between games. Under each game, there are many serves. Each serve contains the serving shots and a few follow-up shots.

In the video shown in **Figure 10**, the first game includes 16 serves. Each serve segment is labeled with the length of the segment, type of play in this serve and the approximate number of strokes in a serve. For example, a label "(L) S B 4" means long segment, server, base-line play and approximately 4 strokes.

All these elements are organized as nodes in a tree. This allows users to navigate from more abstracted top levels to detailed levels. When users click on any nodes, the corresponding key frame will be shown in the right window, and users can start to play the video at the corresponding moment.

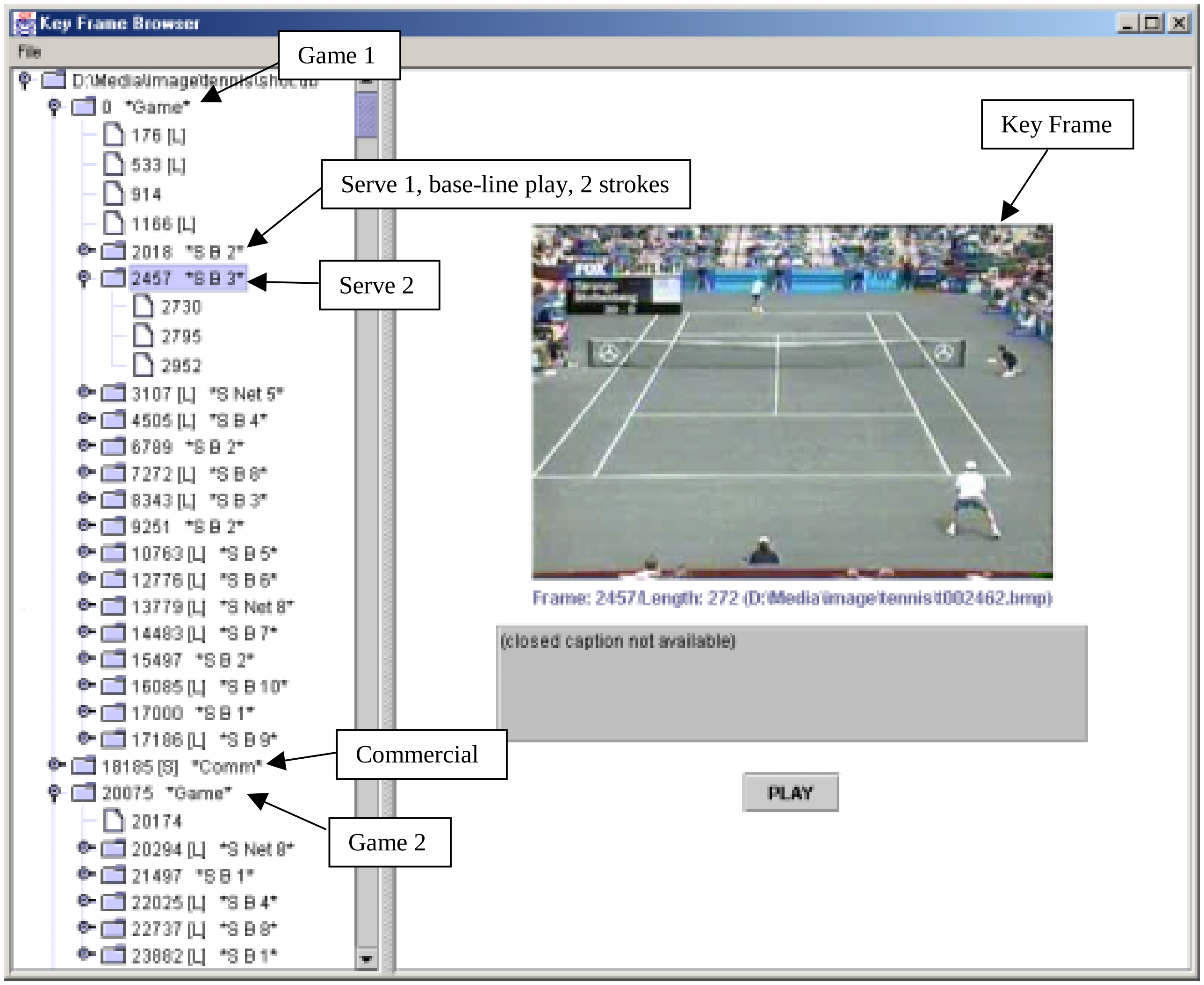


Figure 10 Browsing interface providing hierarchical structure

7 Conclusion and Open Issues

In this chapter, we presented a general framework for structure parsing and high-level event detection for sports videos, using tennis as an example. We have also tested this performance in baseball videos. It combines domain-specific knowledge with automatically segmented objects and features using generic machine learning techniques. Real time processing performance is achieved by exploring redundancy in the compressed video streams. Our experiments have demonstrated high accuracy in specific sports domains such as tennis and baseball.

Under the framework we have developed, there are many more issues that can be studied to produce a comprehensive summary of sports videos. For tennis videos, we can include audio information to detect the number of strokes within each serve more accurately. Score boards are common in all broadcasted sports programs. By detecting these text boxes and recognizing scores, we can understand the status of a game. For example, in a baseball game, we will be able to know when there is a new player entering the field. We can also exam object and/or camera motion at or around special scenes (e.g., serve in tennis, pitch view in baseball) to understand if any interesting events happen. In baseball, when a batter hits the ball, the camera will usually follow the flying ball or the running players.

By analyzing structures and detecting interesting events in videos, we will able to provide efficient browsing, searching and summarization functions to users, possibly in real time for live contents. We can also generate short highlights of important portions within a long game.

References

1. M. Bierling, "Displacement Estimation by Hierarchical Block Matching", SPIE Vol 1001, Visual Communication & Image Processing, 1988.
2. A. Hanjalic, R.L. Lagendijk, J. Biemond: *Automated High-Level Movie Segmentation for Advanced Video Retrieval Systems*, IEEE Transactions on Circuits and Systems for Video Technology, Vol.9, No.4, June 1999
3. Rainer Lienhart, Christoph Kuhmunch and Wolfgang Effelsberg, “On the Detection and Recogonition of Television Commercials”, Proc. of IEEE International Conference on Multimedia Computing and Systems, June, 1997, Ottawa, Canada
4. Toshio Sato, Takeo Kanade, Ellen K. Hughes, Michael A.Smith, “Video OCR for Digital News Archives”, Proc. Of the 1998 International Workshop on Content-based Access of Image and Video Database, January 3, 1998 Bombay, INDIA
5. Drew D.Saur, Yap-Pen Tan etc. “Automated Analysis and Annotation of basketball Video”, Proceedings of SPIE's Electronic Imaging conference on Storage and Retrieval for Image and Video Databases V, Feb 1997.
6. G. Sudhir, John C.M. Lee and Anil K. Jain, “Automatic Classification of Tennis Video for High-level Content-based Retrieval”, Proc. Of the 1998 International Workshop on Content-based Access of Image and Video Database, January 3, 1998 Bombay, INDIA
7. H. J. Zhang et al., “Automatic Parsing and Indexing of News Video”, Multimedia Systems, 2 (6), pp. 256-266, 1995.