# Multi-level analysis of sports video sequences

Jungong Han[a], Dirk Farin[a] and Peter H. N. de With[a,b]

[a]University of Technology Eindhoven, 5600MB Eindhoven, The Netherlands
[b]LogicaCMG, RTSE, PO Box 7089 5605JB Eindhoven, The Netherlands
jg.han@tue.nl

## ABSTRACT

We propose a fully automatic and flexible framework for analysis and summarization of tennis broadcast video sequences, using visual features and specific game-context knowledge. Our framework can analyze a tennis video sequence at three levels, which provides a broad range of different analysis results. The proposed framework includes novel pixel-level and object-level tennis video processing algorithms, such as a moving-player detection taking both the color and the court (playing-field) information into account, and a player-position tracking algorithm based on a *3-D* camera model. Additionally, we employ scene-level models for detecting events, like service, base-line rally and net-approach, based on a number *real-world* visual features. The system can summarize three forms of information: (1) all court-view playing frames in a game, (2) the moving trajectory and real-speed of each player, as well as relative position between the player and the court, (3) the semantic event segments in a game. The proposed framework is flexible in choosing the level of analysis that is desired. It is effective because the framework makes use of several visual cues obtained from the *real-world* domain to model important events like service, thereby increasing the accuracy of the scene-level analysis. The paper presents attractive experimental results highlighting the system efficiency and analysis capabilities.

**Keywords:** sports video, content analysis, moving object, 3-D modeling, feature extraction.

## 1. INTRODUCTION

Sports broadcasts constitute a major percentage from the total video content that is provided by public and commercial television channels. With the advent of more TV channels providing full coverage of large sports events and the continuously increasing recording possibilities, the ways to organize and process such huge data volumes become more important. Moreover, the fact that the users (applications) with different preferences have various requirements makes this problem more difficult. For example, a sports-video storage system may need a content-analysis system to remove all non-playing frames (idle time and commercial advertisement) from the video sequence for efficiently using storage space. However, more detailed information like the moving trajectory of the player, as well as several calculated parameters like real-world speed and running distance of the player in a game might be more useful to a coach and the player. Differently, detection of important events like service makes it possible to deliver sports video also over narrow-band networks, such as Internet and wireless, since the valuable semantics generally occupy only a small portion of the whole content. Therefore, a sports-video analysis system needs to be completely automatic, operate in real or near real-time, and the analysis results should be meaningful for different users (applications). In this paper, we propose a tennis video processing and analysis framework that satisfies these requirements.

Significant research has been devoted to the content analysis of sports videos in the past few years, which can be roughly divided into two classes. Earlier publications[1,2] have only focused on pixel and/or object-level analysis, which segments court lines and tracks the moving players. Apart from the disadvantage that such systems cannot provide the semantic meaning of a sports game, algorithms like player tracking are not robust in the case of a moving camera. Recently, for understanding of the sports game, researchers have paid more attentions to event-based content analysis for sports video.[3–8] Object color and texture features are employed to generate highlights and to parse TV soccer program.[3] Object motion trajectories are used for football-play classification and event detection.[4] However, the resulting object trajectories are obtained in the image domain, which cannot exactly describe the behavior of the players. Similarly, camera motion and some object-based features are employed to detect certain events in soccer video.[5] Unfortunately, unlike a fully automatic system

as proposed in this paper, the object-based features are manually annotated. Kijak *et al.*[6] first define four types of view in tennis video, involving global, medium, close-up and audience, and then detect events like first-service failure in terms of the interleaving relations of these four views. This shot-based model does not take object behavior into account, so that it is not able to provide sufficient classification capabilities. Chang *et al.*[7] utilize the relative position between the court edge and the players to extract service events of a tennis game. Rea *et al.*[8] also extract service events, but based on position and moving area changes of the player information, which provides a more accurate result. Due to the fact that only one or two features are used, such systems cannot provide more advanced semantic analysis. Generally speaking, each system introduced above, in particular tennis video analysis systems, facilitates only one user (application), so that it cannot provide services for multiple users having different preferences. In addition, most existing visual analysis systems only detect one or two specific high-level events, because visual cues are extracted from the image domain. To obtain better analysis results, more real-word visual cues are required. This involves also 3-D modeling of the scene based on reconstruction of camera parameters.

In this paper, we propose a *flexible* framework for *automatic*, *real-time* (or near real-time) tennis video analysis, which can summarize tennis video at three levels in order to take different requirements of users (applications) into account.* The main contributions are as follows.

1) We propose a new player-detection algorithm, which summarizes several effective visual properties in the tennis video (e.g. uniform court color) to build a background, thereby achieving more accurate segmentation results. To make the algorithm more robust, we also add a shadow-detection technique into our segmentation algorithm.

2) We introduce an adaptive Double Exponential Smoothing (DES) filter to track moving players. This method firstly transforms the player positions detected in the image domain into the 3-D domain, based on a camera-calibration algorithm. Subsequently, we propose an adaptive adjustment of filter parameters depending on the real-world speed of the player, which gives better tracking results.

3) We not only extend two common visual cues (e.g., player speed and player position) used by existing sports analysis systems to the 3-D domain, but also propose two novel visual features for modeling important events, like service and net-approach, from a camera viewpoint. The proposed models are capable of classifying each tennis play into three semantic categories, which are popular and familiar to most viewers.

4) Finally, we propose a *flexible* framework for tennis video summarization that enables three-level analysis with a scalable complexity. For instance, object-level and scene-level analysis can be omitted when the pixel-level analysis results are sufficient.

In the sequel, we first give a system overview in Section 2 and then describe the proposed pixel-level algorithms in Section 3. Section 4 and Section 5 introduce object-level and scene-level analysis, respectively. The experimental results on tennis video sequences are provided in Section 6. Finally, Section 7 draws conclusions.

## 2. OVERVIEW OF PROPOSED TENNIS VIDEO ANALYSIS SYSTEM

As mentioned earlier, we attempt to analyze the tennis video sequences at pixel, object and scene level, and further organize them in a scalable fashion. That is, each level not only provides services for users (applications) with different intentions, but also inputs visual cues extracted from this level to the subsequent levels for further analysis.

Figure 1 shows the flowchart of the proposed system, which consists of five modules.

1) *Playing-frame detection.* A tennis sequence, not only includes scenes in which the actual play takes place, but also breaks or advertisements. Since only the playing frames are important for the subsequent processing, we want to efficiently extract the frames showing court scenes for further analysis.

2) *Court detection and camera calibration.* Court information, including size, shape and location, is an important aid to analyze the tennis game. To deduce the semantic meaning from the position and movements of

*This research is part of the European ITEA Candela project.

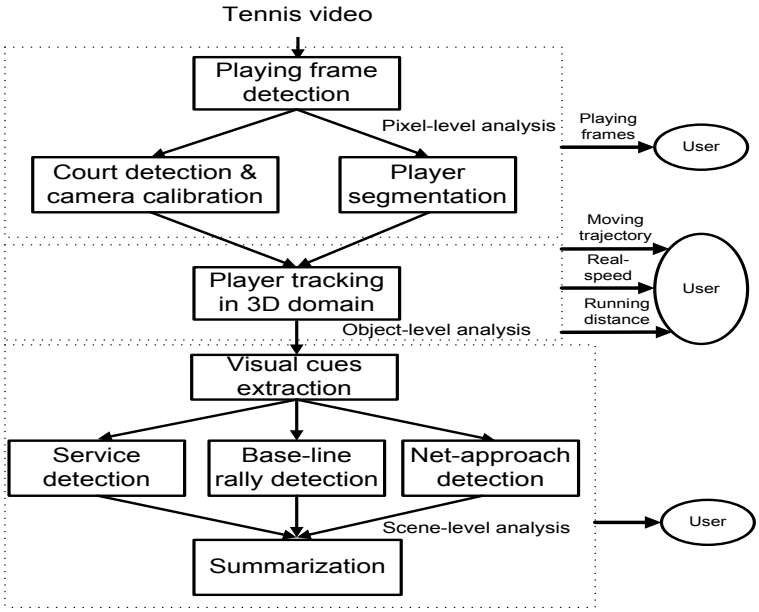


**Figure 1.** Flowchart of the system.

the players, their position has to be known in real-world coordinates. However, pixel-level image-processing algorithms will only give the player positions in image coordinates, which are physically meaningless. To transform these image coordinates to physical positions, a camera-calibration algorithm has to be applied that uses the lines of the court as references to compute the camera parameters.

3) *Moving player segmentation.* The position in the image domain of the player is definitely important to object-level analysis.

4) *Player tracking in 3-D domain.* This module supports more accurate tracking results, and it extracts several real-world visual features for subsequent scene-level analysis.

5) *Scene-level event classification.* This unit first selects several visual cues that enable to describe important events at high level. Then it uses these visual cues and also game-specific contextual information to recognize the events of the tennis game.

In this paper, the first three modules are classified as pixel-level analysis, since visual features used by them are at the pixel level, like color and texture. The fourth module intends to investigate the behavior of moving players, so that it is classified as object-level content analysis. The fifth module aims at extracting the semantically meaningful events, thus it is representing scene-level analysis.

## 3. PIXEL-LEVEL ANALYSIS

This section explains how to analyze the tennis game and extract object-level visual cues, such as the location of the court and the position of the player, using pixel-level features. Since both court detector and player detector rely on accurate detection of the playing frames, we start by presenting our robust playing-frame detection algorithm.

### 3.1. White-pixel Ratio-based Playing Frame Detection

This method only identifies the white pixels of court lines and distinguishes the difference between the number of white pixels of two consecutive frames. We use this metric, because we found that the color of the court line is always white, irrespective of the court type, and the number of white pixels composing the court lines is relatively constant over a large interval of frames (several hundreds). Compared to conventional techniques[1, 2]

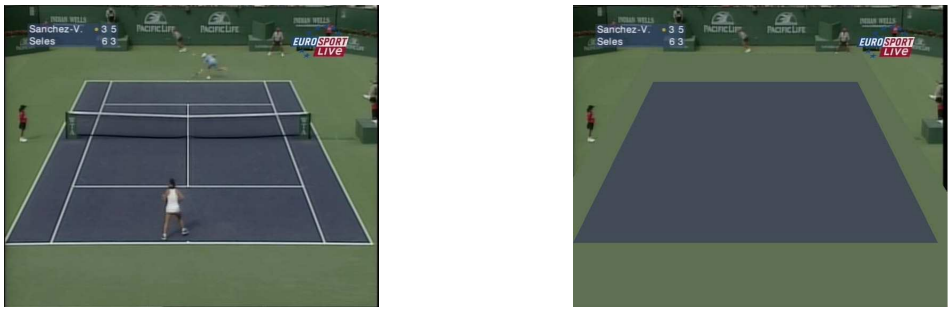


**Figure 2.** An example of constructing a background model.

based on the *mean* value of the dominant color, this technique is more efficient and removes a complex procedure for training data. This algorithm is not explained in this paper, more information can be found in an earlier publication.[9]

### 3.2. Court Detection and Camera Calibration

This algorithm starts by detecting a set of lines. Based on them, we can also compute their intersection points, which provide good features to determine the calibration parameters. The complete system comprises the following algorithmic steps.

1. **Court-line pixel detection.** This step extracts the pixels that belong to court lines. Since court lines are usually white, this step is essentially a white-pixel detector. The speciality of this step is that white pixels that do not belong to court lines (the player's white clothing, the audience, the stadium) are not extracted.

2. **Line-parameter estimation.** Starting with the detected white pixels, line parameters are extracted. We apply a RANSAC-based line detector, which hypothesizes a line using two randomly selected points. If the hypothesis is verified, all points along the line are removed and the algorithm is repeated to extract all the dominant lines in the image. We also determine the extent of the line, such that we obtain line segments instead of infinite lines.

3. **Model fitting.** After a set of lines has been extracted from the image, we need to know which line in the image corresponds to which line in the court model. It may also happen that more lines are detected than are present in the model, or that some of the lines are not detected. This assignment is obtained with a combinatorial optimization, in which different configurations are evaluated.

4. **Court tracking.** When the initial position of the court is known, the computation in successive frames can be carried out more efficiently, since model lines can be assigned to the lines that are closest to the predicted position.

At the algorithm start, Steps 1-3 are carried out to find the initial location of the court in the first image. For the subsequent frames, only Steps 1 and 4 are applied. More details can be found in an earlier publication of our work[10] .

### 3.3. Moving-Player Segmentation

To analyze a tennis video at a higher semantic level, it is necessary to know where the players are positioned. Earlier systems propose several moving-player segmentation algorithms. A class of methods is based on motion detection,[1, 11] in which subtraction of consecutive frames is followed by thresholding to extract the regions of motion. Obviously, with such a detection algorithm, it is impossible to resolve problems where the background is also moving or the camera is moving at the same time. Another category proposes the use of change-detection algorithms. In change-detection systems, the background is first constructed, and subsequently, the foreground objects are found by comparing the background frame with the current video frame. The literature addressing tennis analysis[8, 12] concentrates on selecting a video frame of the tennis court without any players as a background image and then segmenting the players in the video sequence by looking for variations within the background.

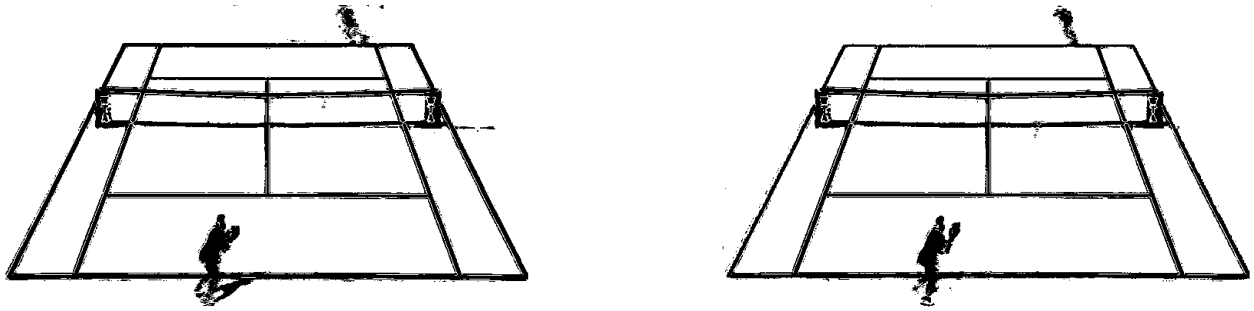

**Figure 3.** An example of constructing the binary map. Left: the binary mask produced after the third step. Right: binary mask produced after the fourth step.

Unfortunately, in most tennis videos, such frames rarely occur. In conclusion, earlier systems adopt existing techniques of moving-object detection without any exploitation of specific properties of the tennis video game, which leads to a poor detection performance. In addition, the purpose of detecting players is to obtain the player's positions. That is, only the feet positions of the players are really important for further analysis, but there is no technique addressing this feature specifically.

The contribution of this paper is also used on change detection, but we focus on building a high-quality background based on the game properties of tennis, since the performance of the change-detection technique depends largely on the quality of the background. We have found that in most tennis video sequences, a regular frame containing the playing field mainly includes three parts: the court (playing-field inside the court lines), the area surrounding the court and the area of the audience. Normally, the moving area of the players is limited to the field inside the court and partially the surrounding area. Moreover, the color of the court is uniform, as is also the case for the surrounding area. The above observations have been exploited to separately construct background models for the field inside the court and the surrounding area, instead of creating a complete background for the whole image. Using this concept, two advantages occur as compared to the conventional algorithms. First, a background picture with better quality is obtained, which cannot be influenced by camera motion. Second, because of the improved background picture quality, only color and spatial information are considered for further feature extraction, which makes our proposal simpler than advanced motion-estimation methods. The complete algorithm is described below.

1. **Construct background for the playing field inside the court.** Up to now, we have obtained the boundaries of the court, and the coordinates of each white pixel of the court lines. We can therefore label the pixels excluding the white pixels within this area as *inside pixels*. After this, it is easy to compute the *mean value* $\mu$ of each color channel and also the *variance* $\sigma^2$ for each color channel. The background model for the playing field inside the court is made, in which the color of each pixel equals the mean color of all *inside pixels*. The stored color background model for an *inside pixel* is $[\mu_r, \mu_g, \mu_b, \sigma_r^2, \sigma_g^2, \sigma_b^2]$.

2. **Construct background for the area surrounding the playing-field border lines.** We predict the playing area for the players outside the court lines, and label all pixels of this area as *outside pixels*. In order to obtain a more robust detection of the surrounding playing area, we predict it using a standard court model constructed in the real-world, which is then transformed into the picture domain employing 3-D camera parameters. Once all the *outside pixels* have been extracted, the same averaging technique as mentioned above is applied to construct the background for the area surrounding the playing-field border lines. Figure 2 shows an example, where the left picture is the original and the right picture is the background built by our algorithm.

3. **Produce binary map.** Create a binary map having the same size as the original picture, and initialize all pixels with 255. This map is used later for player-position analysis. A residue picture is formed by

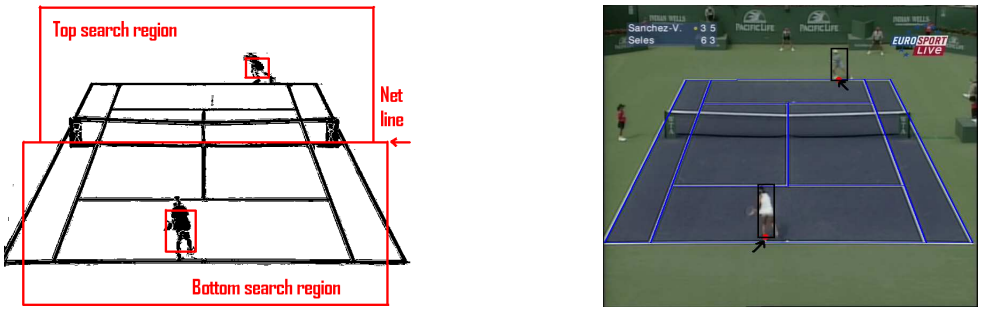


**Figure 4.** Player detection. Left: sketch map for player segmentation. Right: obtained search result (the arrow point at the lower foot position of the player).

subtracting the background model from the original picture. The output binary map is obtained by

$$B(x,y) = \begin{cases} 0, & \text{if } |r-\mu_r| > 3\sigma_r \text{ or } |g-\mu_g| > 3\sigma_g \text{ or } |b-\mu_b| > 3\sigma_b; \\ 255, & \text{otherwise,} \end{cases} \quad (1)$$

where $B(x,y)$ represents the value of the binary map at a given point $(x,y)$, and $r$,$g$,$b$ denote the R,G,B values of the current pixel, respectively. The left picture of Figure 3 shows a binary map resulting from applying Equation 1. A drawback of this method is that it still labels the shadow area as foreground, which will influence the subsequent processing.

4. **Refine binary map.** The purpose of this step is to remove the shadow area from the initial binary map. An area cast into shadow often results in a significant change in intensity without much change in chrominance. This observation has been exploited by previous authors,[13] who labeled the shadow pixels as those pixels that are darker without significant chrominance change. Therefore, we assume that any significant intensity change without significant chrominance change could have been caused by shadow. The normalized chrominance is computed as[13]

$$r_c = r/(r+g+b), \quad (2)$$

$$g_c = g/(r+g+b). \quad (3)$$

This specification enables to compute $r_c$ and $g_c$ for each pixel marked as zero value in the binary map. It is also possible to compute the mean of $r_c$ and $g_c$ for the *inside pixels* and the *outside pixels*. If $|r_c - \mu_{rc}| > Th$, or if the similar test for $g_c$ is true, then the pixel is set to shadow pixel. This produces a mask which has been shown in the right picture of Figure 3. It can be seen that this provides a reasonably good segmentation.

5. **Extract the players in the first frame.** Exploiting the knowledge of the game court, we select two *search regions* in the binary map, above and below the tennis net-line (see the arrow in the left picture of Figure 4). Subsequently, we search the top and the bottom regions of the map above and below the tennis net-line, respectively, for the player bodies. Each player window is specified by the width $W$ and height $H$. These parameters are labeled with a 'B' for bottom and a 'T' for top region, respectively. For example, when searching in the bottom region, we place the center position of the player window of size $W_B \times H_B$ at every possible pixel of the bottom region. For each pixel position, we count the number of zero values in the bottom region of the mask enclosed by the player window at that position. We finally adopt the position that maximizes the zero count. Similarly, we obtain the top-player position using the same algorithm. As mentioned above, the standing position of the player is the most important for our further analysis, because only this point represents the physical position of the player in the real-world. In our case, we used the foot with the lower vertical position as the reference point for the player's position. In practice, shadow pixels act as a kind of noise and blur the exact definition of the player position. Since the shadow pixels have been removed from the binary map, we move up or down the bottom of the player

window to refine its exact position. This step is necessary because we start with a fixed player window size to simplify the player body search. The refinement algorithm is that the bottom is lowered when a player-connected zero component is found in the neighboring row of pixels at the bottom of the player window. Similarly, the bottom position may be moved up when no player-connected zero components are found in the bottom row of the player window. The refinement of the bottom position of the player window gives a more accurate position of the real position in the 3-D model. Figure 4 portrays an example of player segmentation using our algorithm (the left picture) and the segmentation result (the right picture).

6. **Extract the players in the successive frames.** When the initial player positions are determined in the first frame of a court-view sequence, we can detect the players in an efficient way for successive frames. We only explore two local windows surrounding the identified small window centers of the previous frame. Within those two local windows, we again search for the positions that maximize the zero counts in the top and bottom player windows. The feet position of the player is detected in the same way as above.

## 4. OBJECT-LEVEL ANALYSIS FOR VISUAL FEATURE EXTRACTION

The player's behavior, expressed by parameters such as moving trajectory, speed and position, is very useful when the coach and the player want to further analyze the match. Also, in order to analyze the content at a higher semantic level, we need to extract several visual cues as mentioned above. In order to realize this, it is required to measure and track the player's motion. In contrast with conventional sports-analysis systems[4, 8] that track moving objects in the image domain, we track players in the 3-D domain based on a camera-calibration algorithm. There are two advantages. First, tracking in the 3-D domain can yield a good result, which will be shown in the remainder of this paper. Second, for some specific measurements like speed, only visual cues obtained from the real-world domain can be used to quantify the player's behavior accurately. Evidently, it also adds to the analysis of the tennis game at scene-level.

Laviola[14] adopted the Double Exponential Smoothing (DES) operator to track moving persons, which executes approximately 135 times faster than the popular Kalman-filter-based predictive tracking algorithm with equivalent prediction performance. The DES smoothing operator is defined by

$$\begin{cases} s_t = \alpha\ o_t\ +\ (1-\alpha)\ (s_{t-1} + b_{t-1})\ , \\ b_t = \gamma\ (s_t - s_{t-1})\ +\ (1-\gamma)\ b_{t-1}\ ,\ \text{for } t = 3, 4, ... \end{cases} \tag{4}$$

where $o_t$ is the observed position value. The parameter $s_t$ refers to the position value after smoothing the observed position, $b_t$ represents the trend of the player position change, and $\alpha$ and $\gamma$ are two weighting parameters controlling motion smoothness. Note that Equation 4 applies to both the $x$ and $y$ positions of the player in the real-world model, the labeling has been omitted for simplicity. The first two positions of the player are used to initialize the Equation 4, that is, $s_1 = o_1, s_2 = o_2$ *and* $b_2 = s_2 - s_1$. The first smoothing equation adjusts $s_t$ directly for the trend of the previous period with $b_{t-1}$, by adding it to the last smoothed value $s_{t-1}$. This helps to eliminate possible position discontinuities. The second smoothing equation updates the trend, which is expressed as the weighted difference between the last two position values. A key issue is the determination of the two weighting parameters $\alpha$ and $\gamma$. Laviola[14] has proposed to determine them by training and obtain an experience value for those parameters. Obviously, this is not a good solution, because the behavior changes with the player. Therefore, we propose in this paper to derive the values from the real speed measurement, as we track the player in the 3-D domain. The real-speed of the player enables an adaptive adjustment of the two filter parameters, giving better tracking results. The algorithm is carried out in two steps.

1. *Transform 2-D coordinates to 3-D coordinates.* In order to track the player in the 3-D domain, we require the observed position of the player in the image domain and the camera calibration to map 2-D coordinates to 3-D coordinates. In this paper, the observed position is the lower-feet position of the player provided by the player segmentation algorithm in the image domain. The camera model from the court-detection module is used to transform the 2-D position of the player to the physical observed position $o_t$ in Equation 4.

2. *Track player using an adaptive DES filter.* Once we have obtained $o_t$ and $s_{t-1}$, we can compute the real speed of the player. If the speed exceeds a boundary or a significant speed change occurs suddenly, false

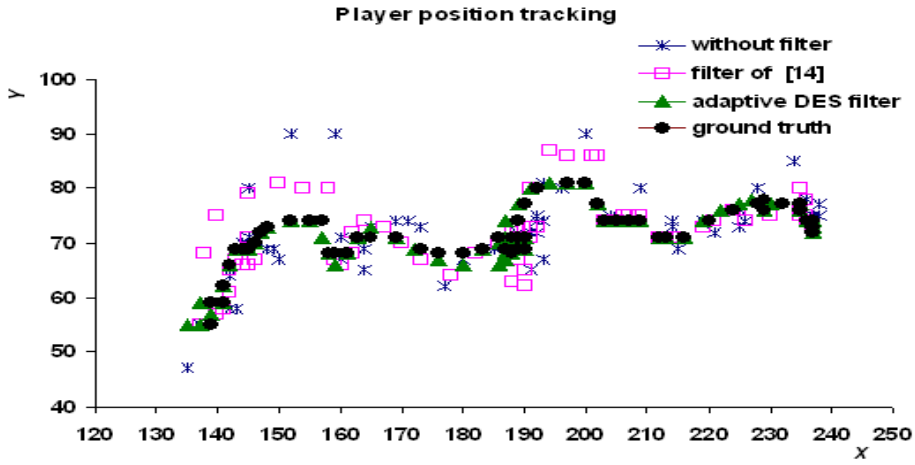


**Figure 5.** Player position tracking, using various filter techniques. X and Y refer to an image domain coordinate system (we track positions in the real-world domain, then transform them to the image domain). The unit of distance is expressed in pixel.

segmentation is likely, as the running speed of a tennis player is normally in the range of 2-7 m/s. Smaller values for $\alpha$ and $\gamma$, occurring when the speed increases, make the current predictive results more rely on the past trend. In this way, we can adaptively control the values of $\alpha$ and $\gamma$ between 0 and 1. This adaptation has given better results than the experience values of the literature. Figure 5 shows examples of the player position tracking with various smoothing filters, where the results of our adaptive DES compares favorably to the manually extracted ground-truth data.

## 5. SCENE-LEVEL ANALYSIS: IMPORTANT EVENTS IDENTIFICATION

Detection and identification of certain events in a tennis game enables generation of more concise and semantically rich summaries. Our proposed system derives several *real-world* visual cues from the previous pixel and object-level analysis and afterwards makes linear models for event recognition. To derive the visual cues, the system should correctly bridge the gap between the numerical image features of moving players and symbolic descriptions of the scene. To do so, we first analyze the game rules and select a list of several visual features that really facilitate semantic analysis of the tennis game. Second, we try to describe each key event making use of the selected visual features from the real-world viewpoint. The previous two steps are performed off-line, and yield mapping and computing models for events. These models are used in the algorithms for on-line computations of both steps. Third, we compute a likelihood degree of each event for each input frame and decide on the mapping of input frames to events. Fourth, a simple but efficient temporal filter is used to extract the start time and the end time of each event. Finally, we summarize the game based on correlations between events.

### 5.1. Real-World Features in Tennis Video

As mentioned earlier, some existing tennis video analysis systems[7, 8] employ two common visual features: position and speed of the player. In this paper, we not only extend these two features to the real-world domain, but also propose two novel features for event identification in tennis video, which makes it possible to detect more events. Let us now list and motivate the four real-world visual features that are used by our algorithm.

- **Instant Speed of the Player:** The speed of each player is definitely important, because it reveals the current status of a player (running or still) and it also indicates the intensity of the match.
- **Speed Change of the Player:** Acceleration and deceleration of a player occurs during changes in action behavior.

**Figure 6.** Frames extracted from three different events. Left: service. Center: base-line rally. Right: net-approach.

- **Relative Position of the Player to the Court Field:** This position is instrumental for the recognition of those events that are characterized by a typical arrangement of players on the playing field.
- **Temporal Relations among Each Event:** In some sports games like tennis and baseball, there are strong temporal correlations among key events. For example, in a tennis video, service is always at the beginning of a playing event, while the base-line rally may interlace with net-approaches.

### 5.2. Visual-based Model for Each Event

With the above real-world cues, we can model key events, of which three are given below (Figure 6 shows a select frame from three different events).

- **Service:** This event normally starts at the beginning of a playing event, where two players are standing on the opposite half court, and where one is at the left part of the court, and the other is at the right part. In addition, the receiving player has limited movement during the service.
- **Base-line Rally:** The base-line rally occurs usually after the service, where two players are moving along their base-lines with relative smooth speed, that is, there is no drastic speed change.
- **Net-approach:** This is one of the highlight parts of a game, in which standard visual cues are a large speed change combined with close positioning of players to the net lines.

All event models utilize the four real-world visual features described earlier. We have found that a linear combination of these visual features can be applied to decide what kind of events the current frame belongs to.

### 5.3. Event Extraction

Up to now, each frame is classified as the service, base-line rally or the net-approach. The next step is to extract the start time and the end time of each event. Figure 7 gives a practical example, in which we show a set of frames in the temporal direction. A circle represents a detected "service" frame, and a cross stands for a "non-service" frame. It can be noted from Figure 7 that the first frame is not a suitable start frame of the service event, although it is labeled as a "service" frame. This is because there are three "non-service" frames behind it, so that the probability that it is a wrong "service" frame is large. Thus, the first step of start-time extraction is to measure the local correlation of the $i$-th frame using the following definition:

$$c(i) = s(i-2) + s(i-1) + s(i) + s(i+1) + s(i+2), \tag{5}$$

where $i$ is the frame number, and $s(i)$ denotes the state of this frame. This state is defined by

$$s(i) = \begin{cases} 1, & \text{if } i\text{-th frame is signed as "service";} \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

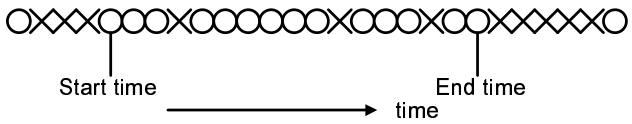


**Figure 7.** An example to show how to extract the start time and the end time of the service event.

We compute the local correlation $c(i)$ for each "service" frame (circles), then select the minimum $i$ with $c(i) > 2$ as the start frame of the service event.

In order to detect the last frame of the service, we first calculate the occupancy rate by

$$O_i = (f_i - f_1)/N_t. \tag{7}$$

Here, $f_i$ is the frame number of the current "service" frame, and $f_1$ is the frame number of the start "service" frame. Parameter $N_t$ denotes the amount of image frames between $f_i$ and $f_1$. The frame with the maximum $i$ with $O_i > 0.7$ is selected as the end frame of the service.

### 5.4. Game Summary

Our scene-level analysis not only identifies some important events, but also intends to summarize the game, making use of correlations between events. For instance, if there is a service event without a base-line rally or a net-approach that directly changes to a "no-event", it is reasonable to deduce that such a case might be an ace or a double-fault. Furthermore, it is feasible to calculate how many net-approaches each player carried out during a match. Based on this value, the player with more net-approaches is classified as more aggressive.

## 6. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed algorithm, we tested our system using several broadcasted tennis videos recorded from three different tennis matches. The first one is collected from the US open (15 minutes, 720×576, 25 frames/second), the second one is from the Australian open (6 minutes, 320×240, 25 frames/second), and the third sequence is from the French open (4 minutes, 720×576, 25 frames/second).

### 6.1. Results for Pixel and Object-Level Algorithms

We evaluated our playing-frame detection algorithm, court-detection algorithm and player segmentation and tracking algorithm. The system achieves a 98% detection rate on finding court-view frames, 98% rate on court detection and camera calibration, and 96% detection of players, where the criterion is that at least 70% of the body of the player is included in the detection window. Figure 8 gives a set of practical detection results. It can be concluded from these results that our proposed algorithm not only accurately segments the player and the court, but also detects the position of the player in the image domain.

### 6.2. Results for Scene-level analysis Algorithms

We manually labeled all events to obtain the ground truth. Table 1 shows the results of our scene-level analysis simulations. It can be concluded that the scene-level event extraction rate of the system is about 90%.

Figure 9 shows the user interface of our tennis video analysis software.[†] From this user interface, the viewer can find analysis results at three different levels. At the pixel level, several key objects -both moving and still- are segmented and indicated. Meanwhile, the system indicates whether the current frame is a court-view frame or not. At the object level, the moving objects are tracked in the 3-D domain (at the right side). At the scene level, the system automatically classifies three important events. Our system is efficient, achieving a real-time or near real-time performance (2-3 frames/second for 720×576 resolution, and 5-7 frames/second for 320×240 resolution, with a P4-3GHz PC).

[†]More results can be found on http://vca.ele.tue.nl/demos.

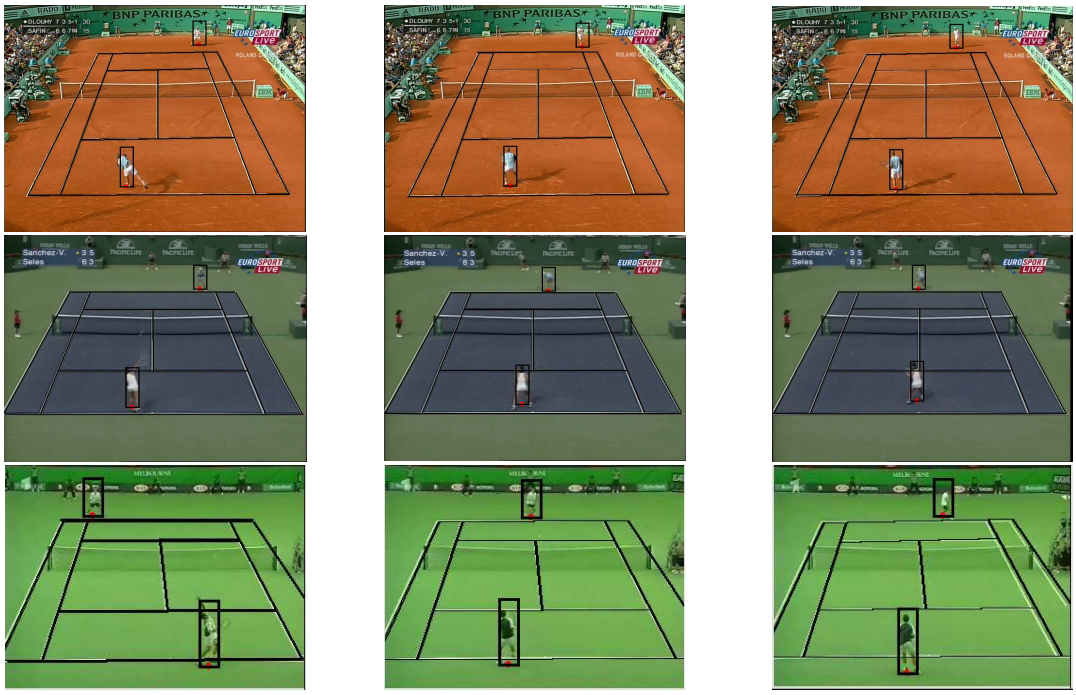

**Figure 8.** Player and court-tracking results for 3 consecutive periods of 30 frames (the court is indicated by black lines, the black rectangular represents the detected player). Top row: tennis video sequence recorded from the French open. Center row: tennis game recorded from the US open. Bottom row: tennis game recorded from the Australian open.

**Table 1.** Classification results.

| | Detect | Correct | Miss | False |
|---|---|---|---|---|
| Service | 21 | 21 | 2 | 0 |
| Base-line Rally | 16 | 14 | 0 | 2 |
| Net approach | 8 | 7 | 0 | 1 |

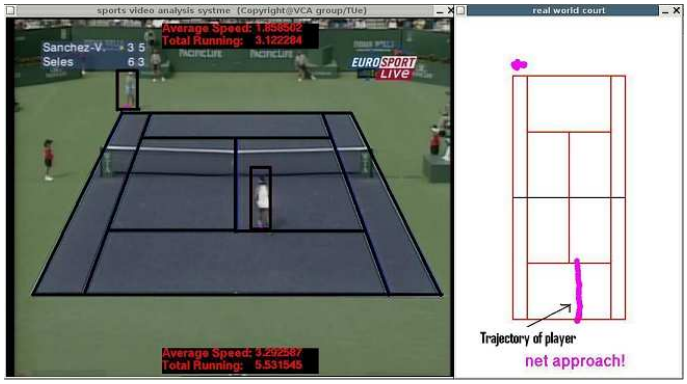


**Figure 9.** Results of our analysis system. The left image shows the detected court and players, as well as the average speed of each player (top and bottom). At the right is the real-word court model, where the trajectory of each player is visualized, in this case showing a net-approach.

## 7. CONCLUSIONS AND FUTURE WORK

In this paper, a new framework for analysis of tennis video has been introduced. The proposed framework enables the analysis and semantically meaningful summarization of the tennis game at three levels according to user preferences. The system includes several novel pixel-level analysis algorithms like a player-segmentation algorithm featuring a region-dependent player discrimination and a playing-frame detection algorithm based on white-pixel fraction occurring in the playing field. Additionally, we have deployed a number of novel models for important events based on real-world visual cues. The new algorithms showed a detection rate and/or accuracy of 90-98%. Our system was integrated into a networked home multimedia application as a video-content analysis unit, which was successfully demonstrated at an international multimedia conference.[15]

We have seen that the player segmentation and tracking steps are the most time-consuming part of our system. This can be reduced significantly by a eliminating the background reconstruction for every processed frame. In the near future, we will include audio-analysis functions into our sports content-analysis system. Also, the system may be extended to analyze other sport types, like volleyball, badminton and basketball, since several techniques, such as court detection, camera calibration, player segmentation and high-level analysis, are generally applicable.

## REFERENCES


1. G. Sudhir, C. Lee and K. Jain, "Automatic classification of tennis video for high-level content-based retrieval", Proc. IEEE international workshop on content based access of image and video databases, pp. 81-90, 1998.
2. C. Calvo, A. Micarelli and E. Sangineto, "Automatic annotation of tennis video sequences", Proc. DAGM-symposium, pp.540-547, Springer, 2002.
3. Y. Gong, L. Sin, C. Chuan, H. Zhang, "Automatic parsing of soccer programs", in Proc. IEEE Int. Conf. Mult. Comput. Syst, pp.167-174, 1995.
4. V. Tovinkere and R. Qian, "Detecting semantic events in soccer games: toward a complete solution", in Proc. IEEE Int. Conf. Mult. Expo (ICME), pp.1040-1043, Aug, 2001.
5. J. Assfalg, M. Bertini, A. Bimbo, W. Nunziati and P. Pala, "Soccer highlights detection and recognition using HMMs", in Proc. IEEE Int. Conf. Mult. Expo (ICME), pp.825-828, Aug, 2002.
6. E. Kijak, L. Oisel and P. Gros, "Temporal structure analysis of broadcast tennis video using hidden Markov models ", in Proc. SPIE Storage and Retrieval for Media Databases 2003, pp.289-299, January, 2003.
7. S. Chang, D. Zhong and R. Kumar, "Real-time content-based adaptive streaming of sports videos", in Proc. IEEE Workshop Content-based Access to Video Library, pp.139-143, December, 2001.
8. N. Rea, R. Dahyot and A. Kokaram, "Classification and representation of semantic content in broadcast tennis videos ", in Proc. IEEE Int. Conf. Image Processing, Sep. 2005.
9. J. Han, D. Farin, P. H. N. de With and W. Lao, "Automatic tracking method for sports video analysis ", in Proc. Symposium on information theory in the Benelux, Brussels, Belgium, pp. 309-316, May, 2005.
10. D. Farin, J. Han and P. H. N. de With, "Fast camera-calibration for the analysis of sports sequences", in Proc. IEEE Int. conf. Mult. Expo (ICME), July, 2005.
11. G. Pingali, Y. Jean and I. Carlbom, "Real time tracking for enhanced tennis broadcasts", in Proc. IEEE Int. conf. Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), June, 1998.
12. T. Bloom and P. Bradley, "Player tracking and stroke recognition in tennis video", in Proc. WDIC, June, 2003.
13. S. Mckenna, S. Jabri, Z. Duric and A. Rosenfeld, "Tracking groups of people", Computer Vision and Image Understanding, No.80, pp. 42-56, 2000.
14. J. Laviola, "An experiment comparing double exponential smoothing and Kalman filter-based predictive tracking algorithms", In Proc. IEEE Int. conf. virtual reality, pp. 283-284, 2003.
15. Jan Nesvadba *et al.*, "Real-time and distributed AV content analysis system for consumer electronics networks", in Proc. IEEE Int. conf. Mult. Expo (ICME), July, 2005.