Tennis Real Play

Jui-Hsin Lai, Chieh-Li Chen, Po-Chen Wu, Chieh-Chi Kao, Min-Chun Hu, *Member, IEEE*, and Shao-Yi Chien, *Member, IEEE*

***Abstract*—Tennis Real Play (TRP) is an interactive tennis game system constructed with models extracted from videos of real matches. The key techniques proposed for TRP include player modeling and video-based player/court rendering. For player model creation, we propose the process for database normalization and the behavioral transition model of tennis players, which might be a good alternative for motion capture in the conventional video games. For player/court rendering, we propose the framework for rendering vivid game characters and providing the real-time ability. We can say that image-based rendering leads to a more interactive and realistic rendering. Experiments show that video games with vivid viewing effects and characteristic players can be generated from match videos without much user intervention. Because the player model can adequately record the ability and condition of a player in the real world, it can then be used to roughly predict the results of real tennis matches in the next days. The results of a user study reveal that subjects like the increased interaction, immersive experience, and enjoyment from playing TRP.**

***Index Terms*—Image-based rendering, sports game, tennis video, video rendering, video-based rendering, wiimote control.**

I. Introduction

In recent years, a number of interactive videos have been proposed on the Internet. For example, there are now several videos of magic shows on YouTube. These videos are unique in that users not only watch the videos but also participate in the show to guess the answers to magic tricks. It is our opinion that such interactive videos can engage user interest, and that the concept of interactive videos can be further extended. In this paper, we propose Tennis Real Play (TRP), an interactive tennis game constructed from videos of real matches. As shown in Fig. 1, TRP includes three technical components: video analysis/annotation, video-based rendering, and interactive sports games. Previous relevant work is reviewed below.

A. Video Analysis/Annotation

Like event annotation and content segmentation, video analysis/annotation is a popular and important topic because of the dramatic increase in the number of videos. Many previous studies on this issue have been published. For extraction of sport videos, Wang *et al.* [1] proposed a method to analyze video structure and automatically replay highlights of soccer videos. Wang and Parameswaran [2] analyzed the ball trajectories in tennis matches to detect player's hitting tactics for video annotation, and Zhu *et al.* [3] proposed recognition of player actions in tennis videos for event annotation. For content segmentation of tennis videos, Lai *et al.* [4] proposed methods for separating and integrating content to enrich videos of tennis matches. By applying these previously studied methods, a large number of videos can be organized so that users can immediately find highlights in a long video.

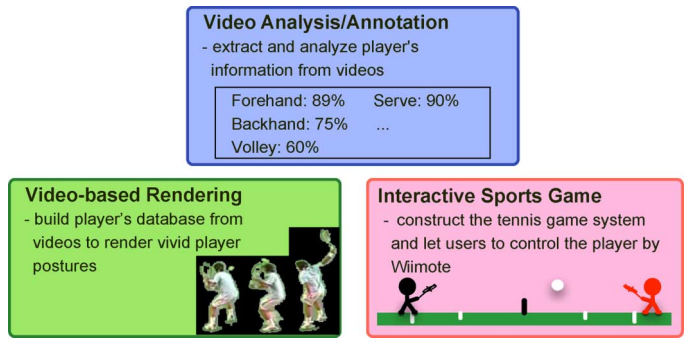

Fig. 1. Tennis Real Play includes three technical components: video analysis/annotation, video-based rendering, and interactive sports games.

B. Video-Based Rendering

Video-based rendering, which can be described as an extension of image-based rendering, is a method to rearrange video frames to create a new video. In studies of video-based rendering, Schodl *et al.* [5] proposed video textures to automatically cut videos into clips and re-loop them to create continuous animations. Efros *et al.* [6] proposed methods to recognize and classify player actions in football videos. Using such clips of player actions, new actions can be synthesized. Inamoto and Saito [7] rendered a free-view football game from videos from multiple cameras, with the free-view synthesis yielding a fresh viewing experience. Lai *et al.* [8] proposed Tennis Video 2.0, which rendered multiple players in continuous motion at the same time to create a more interesting viewing experience. Those previous works all provided viewers a novel viewing experience.

C. Interactive Sports Games

Interactive sports games are a popular form of entertainment. In particular, the interactive tennis game in Wii Sports[1] is one

[1]Wii Sports: http://wiisports.nintendo.com/

Manuscript received October 24, 2011; revised February 15, 2012 and April 17, 2012; accepted April 18, 2012. Date of publication May 01, 2012; date of current version November 22, 2012. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Nicu Sebe.

J.-H. Lai, C.-L. Chen, P.-C. Wu, C.-C. Kao, and S.-Y. Chien are with the Media IC and System Lab, Graduate Institute of Electronics Engineering and Department of Electrical Engineering, National Taiwan University, Taipei 106, Taiwan (e-mail: juihsin.lai@gmail.com; chiehli.chen@gmail.com; chiehchi.kao@gmail.com; pcwu@media.ee.ntu.edu.tw; sychien@cc.ee.ntu.edu.tw).

M.-C. Hu is with the Research Center for Information Technology Innovation, Academia Sinica, Taipei 115, Taiwan (e-mail: trimy@citi.sinica.edu.tw).

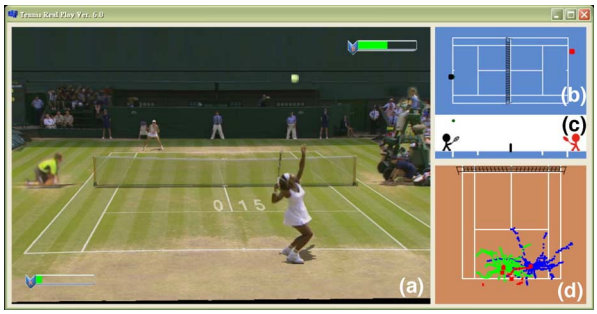

Fig. 2. User interface consists of four components: (a) the main screen of the rendering result, (b) the coordinates of the players and ball on XY-axis, (c) the coordinates of the players and ball on YZ-axis, and (d) motion paths in the player database.

of the most well-known among such games. Innovative user interfaces such as those of Wiimote and Wii Fit provide a more realistic gaming experience and changes the way people play games. Not surprisingly, Wii has successfully engaged users throughout the world. In contrast to the interactivity in Wii, Play Station 3 (PS3) places more emphasis on visual quality. Top Spin 3[2] is one of the tennis games on PS3, and the realistic player postures and lighting effects offer the user a more vivid viewing experience. Inspired by both games, one of the aims of this study is to build a tennis game that offers an interactive experience similar to that of Wii along with vivid video texture.

TRP is an interactive tennis game system constructed with models extracted from videos of real tennis matches. As shown in the game frame in Fig. 2(a), the textures of the players and the background court are extracted from real videos of matches, and player postures are immediately rendered according to the user's control. To implement TRP, we propose a system framework consisting of player model creation and player rendering, and a video with system overview is available on the website.[3] When creating a player model, projection to real world coordinates is proposed to normalize object sizes and the motion trajectories of segmented players. Next, a 4-state-transition model is proposed to model tennis players' behaviors. For player rendering, we propose methods to select suitable clips (move/hit/standby) from the database. The movement abilities and hitting strength of a rendered player will depend on these clips and statistics in the videos of real matches, which is the most interesting and unique feature of player rendering. Subsequently, we construct a 3-D model of the tennis court from the background image and build the game system on this model. As shown in Fig. 2(b) and (c), the player's state is recorded in 3-D coordinates and the game frame is rendered with a virtual camera. By combining the 3-D model with techniques in video-based rendering, the proposed system can render game frames in different viewing angles. The contributions of this paper are listed below.

- To the best of our knowledge, this is the first work to integrate video-based rendering and interactive sports game, and all the rendered characters perform like the players in the real world. By combining the 3-D game model, the rendered characters are even able to compete with hand-drawn characters. The experimental results show that the proposed work gives users a realistic gaming experience and provides them much more fun by interacting with match videos.
- Conventional video games utilize expensive motion capture systems to build the player database which results in rather fixed action sets. Moreover, the player model cannot be updated often. The experimental results reveal that the presented methods can build player database from match videos, which might be a good alternative for motion capture systems. In addition, a new match content can be integrated to the game as soon as a new match video is available without much user intervention.
- The game results of TRP can reflect the match results in real world because the game characters can adequately record the abilities and conditions of players in the match video. From the experiments, this property can then be used to simulate the game results which match the results in the original match videos, and it may be used to predict the results of real matches in the coming days.

In addition, this paper is an extension of our previous work [9]. The additional content includes the methods of player extraction in Section II, method for database reduction in Section III-C, method for signal analysis of Wiimote for gesture recognition in Section V-B, method for moving model of ball in Section V-C, a more complete discussion and results of game rendering in Section VI-A, the extraction results of hitting time detection in Section VI-B, evaluation and comparison for clip transition in Section VI-C, and a more complete subjective evaluation and the demonstration in public of the proposed system in Section VI-F.

II. Player Extraction

A. Generation of Background Court

The generation of background court is the first step of player extraction, which can be used for the player segmentation and be edited as the background model in the game rendering. For previous works in generating background scenes, Lai *et al.* [10] generated background scenes of the input video by projecting video frames to the sprite plane. Here, we employ Lai's method to generate the background scene for tennis court.

As the first step in Fig. 3 shows, the video frames are projected to the sprite plane with the perspective transform. The perspective transform is able to model the possible camera motions in the tennis video including panning, tilting, and zooming. Thus, all the video frames could be projected to the sprite plane. Each video frame is composed of foreground objects, like players or ball boys, and the background scene. By projecting all video frames to the sprite plane, we can generate the scene of tennis court from preserving the pixels belonging to the background scene but foreground objects. The main idea to detect foreground objects in the frame is based on the observation that the moving objects do not occupy a fixed position on the match court for a long time. Thus, we can assume that the maximum temporal distribution on each pixel location in the sprite plane should be the background. Furthermore, the pixel correlation of spatial co-appearance also needs to be considered for determining the background pixels because the foreground objects and background scenes

[2]Top Spin3: http://www.topspin3thegame.com/

[3]Demo videos: http://media.ee.ntu.edu.tw/larry/trp

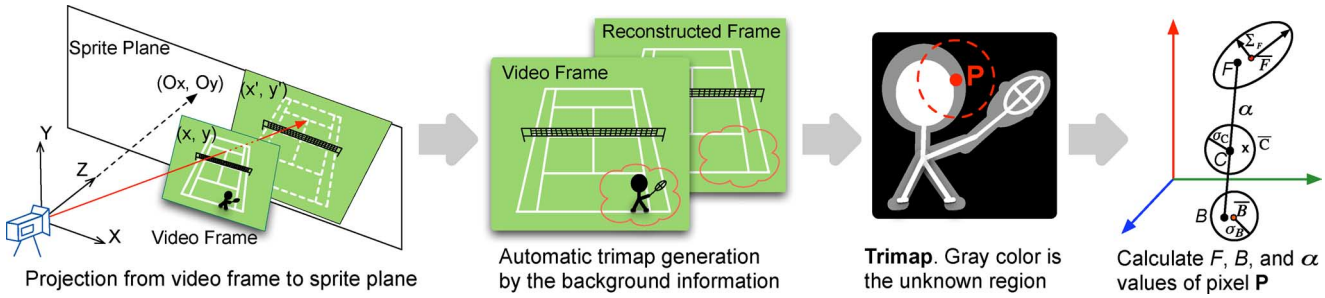

Fig. 3. Processing flow of background projection and foreground segmentation.

sometimes may have equivalent appearance probabilities, like the foreground objects occupying a fixed region for a period of time. The more detailed descriptions of equations can refer to [10], and some results of background generation are available in their website.[4] Next, we will use the background scene for player segmentation and manually label the 3-D information of tennis court as the game scene in the game rendering.

B. Segmentation of Foreground Objects

Many previous works had studied the methods for video segmentation. Here we employ the segmentation method in [8] for foreground objects, like players, ball boys, and the ball in tennis videos. First, the video frame without foreground objects can be reconstructed from the sprite plane, which is also called a reconstructed frame as the second step in Fig. 3. Next, the matting model is employed as the segmentation model. The matting equation in (1) models both the foreground and background color distributions with spatially-varying sets of Gaussians, and assumes a alpha blending of the foreground and background colors to produce the final output:

$$C = \alpha F + (1 - \alpha)B \tag{1}$$

where C, F, and B are pixel's composite, foreground, and background color, respectively, and α is the pixel's opacity component used to linearly blend between the foreground and background. However, the matting procedure needs a trimap as the prior knowledge for segmentation. The trimap as shown in the third step of Fig. 3 is a map indicating the foreground (white), background (black), and unknown (gray) regions in the image. The trimap can be automatically generated without human-assistance by calculating the difference between reconstructed frame and video frame. Next, we use Bayesian matting equations, with background pixel-value modeled from background court, to solve the values of blending factor α and foreground color F of each pixel on the unknown region. The fourth step in Fig. 3 is an illustration of Bayesian matting model with pixel-values in Gaussian distribution. For more detailed threshold settings and descriptions of equations, please refer to [8]. Moreover, some segmentation results are available in the website.[5]

[4]http://media.ee.ntu.edu.tw/larry/sprite

[5]http://media.ee.ntu.edu.tw/larry/tennis2

C. Hitting Detection

In many sports videos, the moment, when the player hits the ball, is a key event and contains semantic information for video analysis, like shots in soccer, strikes in baseball, and rallies in tennis. To detect player's positions or ball trajectories in tennis videos, we need to calculate the parameters of camera calibration first. Several previous works proposed methods for automatic camera calibration and gave some novel applications, like 3-D virtual content insertion [11], virtual advertisement insertion [12], and mixed-reality systems on mobile devices [13]. Here, we refer to the camera calibration method, a perspective transformation from frame coordinate to the real world coordinate, for tennis court in [11]. To calculate the calibration parameters, the ground features like court line intersections are extracted and used to compute the camera calibration. Next, all the positions of segmented foreground objects in frame coordinate are projected to the real world coordinate.

The main idea of detecting the hitting moment is to locate the time index at which player's moving path intersects with the ball's trajectory. The player's moving path can be extracted by projecting the centroid of player's connected component on the ground plane. In contrast to player's moving path, the extraction of ball's trajectory is much difficult. The ball sometimes disappears from the broadcasting video due to its high speed of movement, quantization of video coding, or being occluded by players. Furthermore, there are some false candidates of ball from the segmentation to make the trajectory extraction difficult. First, we use sieves to preserve the possible ball candidates from the segmentation results. The sieves can be designed with size filters, color filters, or location filters [11], but these filters could not solve the problem that the ball is absent from ball candidates. By using the continuous property of ball trajectory, we construct the ball's motion model to predict and complete the trajectory. The Kalman-based prediction model is proposed to extract ball trajectories. The Kalman filter is an efficient recursive filter that estimates the state of a dynamic system from a series of incomplete and noise measurements [14]. We modify the prediction model as follows:

$$\mathbf{x_t} = \mathbf{F_t}\mathbf{x_{t-1}} + \mathbf{w_t} \tag{2}$$

where $\mathbf{F_t}$ is the force for state transition, $\mathbf{w_t}$ is the process noise, and $\mathbf{x_t}$ models the state of the ball's position, velocity, and acceleration at time t. We use the camera calibration model to project the values of ball position, velocity, and acceleration to real

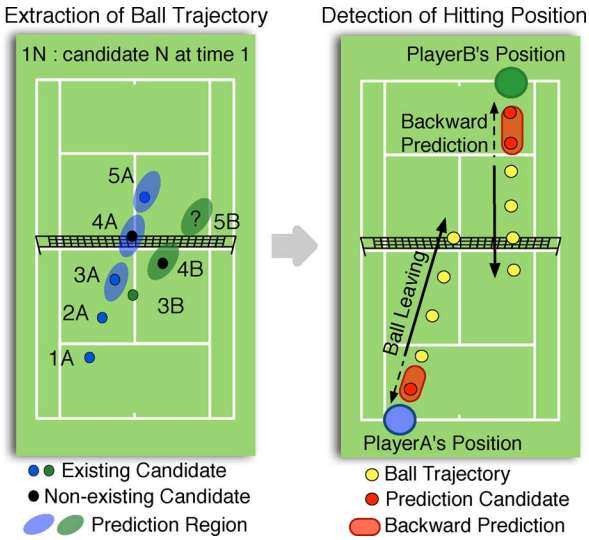

Fig. 4. Illustrations for the extraction of ball trajectory and the detection of player's hitting position.

world coordinates in order to compensate for camera parameters. Actually, the projection values of ball in real world coordinate are not correct because the calibration model only applies for the objects on the plane of court floor. Here we find that the ball's height is less drastically changing during the flying after the hitting, but it is drastically changing when the ball went cross the net or approached to the player, like the bouncing on the court. Thus, we can still project the segment of ball trajectory before crossing the net to the real world coordinate and linearly shift the projection trajectory to compensate ball's height in the flying. Note that the state transition $\mathbf{F_t}$ can be designed with a high-order model to simulate the force of gravity and air friction during the ball flying. The higher order it sets, the more accurate prediction results are extracted.

The first step in Fig. 4 shows an example of prediction and complement of ball trajectory. 1A means one candidate detected at time 1, and 3A and 3B mean two candidates detected at time 3. The prediction model uses the positions of 1A and 2A to model the initial moving state and predict the region at time 3, and the prediction result is positive if one existing candidate located on the region. We can find that two existing candidates in time 3 and a new prediction model would be created to model a new possible ball trajectory. If there is no existing candidate on the prediction region, the non-existing candidate would be set by the prediction model, which would be recognized as a true candidate if one existing candidate existed at the subsequent prediction. The extraction procedure repeats the prediction, and the path with the longest prediction length will be considered to be the ball trajectory. We can see that two candidate trajectories, trajectory [1A – 5A] and trajectory [2A, 3B, 4B], are detected, and only the longer one would be preserved.

Next, the hitting moment is determined by the time index of the interaction of player's moving path and the ball's trajectory. With the prediction model in (2), the ball trajectory can be backward predicted. As shown in the second step in Fig. 4, the yellow dots are the extraction results of ball trajectory and the red dots represent the backward prediction of the movement model. The hitting time can be considered as the time index with a minimum distance between the player's moving path and the ball's trajectory. So, the player position at hitting time is considered as the hitting position, and the hitting direction of each rally is detected by connecting the subsequent hitting positions. The hitting strength of each rally is extracted by the moving speed of prediction model. The statistics of hitting directions and strengths of the player provide a true property in the player rendering.

D. Hit Statistics

With hitting strength and directions of each rally are detected in Section II-C, the hits must be further categorized. The hit categories include forehand volley, backhand volley, forehand stroke, backhand stroke, and drop shot. However, it is difficult to identify each hit category from videos with a single camera view. To simplify the identification process, two hit categories, forehand and backhand, are first labeled by the program. Several previous studies have investigated this problem. For example, Roh *et al.* [15] proposed a curvature scale space (CSS) to describe the characteristic shape features of the extracted players and used it to detect hit categories. However, we find that features in CSS cannot accurately identify hit categories in our experiments because players' shapes are dissimilar, and it is difficult to find common features in the same hit category. Using the methods in [3], we calculate and sum the values of optical flow on the right and left sides of the player. If the optical flow on the right side of the player is larger than that on the left side, the hit posture is regarded as a forehand when the player's handedness is the right. Otherwise, the hit posture is regarded as a backhand. Note that player's habitually using is manually recorded before analysis.

Once forehand and backhand hits are categorized, we use the player position as a clue to identify volleys and strokes. The posture is identified as a volley if the player's position is close to the net and a stroke if player's position is far from the net. Although these assumptions are not always correct, most of the time they are true. In addition, the classification rates between stroke and volley can be improved by more information from different viewing angles of cameras. In a short summary, after detecting the hit time and identifying the hit category, each hit posture is classified as a forehand stroke, backhand stroke, forehand volley, or backhand volley.

III. Player Model

Player model creation is a key component in TRP because player behaviors in the game including hitting preference and movement characteristics are controlled by the model. Each player has a unique player model in different tennis games. Consequently, the essence of this study is the creation of player models from videos of real matches.

A. Database Normalization

The player database not only includes segmentation masks but also records where a player stands in each video frame. Nevertheless, the scale of each mask and position in the player database changes as a result of camera zooming or panning. The

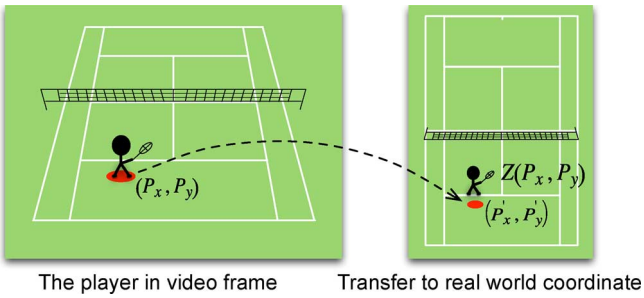

Fig. 5. Illustration of player position and size normalization.

effect of scale change is also known as the perspective transformation. Therefore, a procedure is required to normalize the database. As mentioned in Section II-C, we use a perspective model to project foreground objects from frame coordinate to real world coordinate to compensate the camera parameters. Next, we assume that the bottom coordinates of the player mask (P_x, P_y) specify the position of the feet on the court. The normalized coordinates are presented as (P'_x, P'_y) by projecting (P_x, P_y) onto the real world coordinate system according to the following equations:

$$P_{x'} = G_x(P_x, P_y) = \frac{m_0 P_x + m_1 P_y + m_2}{m_6 P_x + m_7 P_y + 1} \quad (3)$$

$$P_{y'} = G_y(P_x, P_y) = \frac{m_3 P_x + m_4 P_y + m_5}{m_6 P_x + m_7 P_y + 1} \quad (4)$$

where m_0 to m_7 are the parameters of perspective projection. As players are extracted from video frames, the sizes of the segmented players will be influenced by camera parameters and a player's position on the court. Therefore, two factors are considered to normalize player size: the projection parameters of the camera and the player's court position; or it can be simply written as the zooming function of perspective transform with considering the standing position. The zooming rate parameters of the players can be estimated by calculating the deviation of $G_x(P_x, P_y)$ and $G_y(P_x, P_y)$ with respect to the horizontal and vertical zooming factors on the coordinates (P_x, P_y), respectively. Therefore, the normalized player size $Z(P_x, P_y)$ is obtained by multiplying the horizontal and vertical zooming factors, as expressed in the following equation:

$$Z(P_x, P_y) = \frac{\partial G_x(P_x, P_y)}{\partial P_x} \frac{\partial G_y(P_x, P_y)}{\partial P_y}. \quad (5)$$

As shown in Fig. 5, the normalized player position and size can be calculated with perspective parameters and the player's position.

B. Behavior Model

Even after the player database has been constructed and the hit statistics generated, it remains difficult to vividly render a player's behavior without a behavior model. In previous studies of video-based rendering, Schodl and Essa [16] and Colqui *et al.* [17] synthesized new video clips by rearranging frames in the original video sequences. Both studies presented ingenious schemes to create new motions by displaying the original video clips in a different order. However, most of the objects in their test videos have simpler structures such as those of hamsters, flies, or fish. No human videos were used because human behaviors are much complex. For video-based rendering with human motions, Phillips and Watson [18] introduced template sequences and performed two forms of data-based action synthesis: "*Do as I Do*" and "*Do as I Say*". Although the results were impressive, the system could only synthesize actions based on templates of existing motion sequences. Furthermore, Flagg *et al.* [19] presented photo-realistic animations of human actions. By pasting numerous sensors on a human subject's body to detect motion, this study overcame many challenges encountered in synthesizing human actions and achieved vivid rendering effects. Nevertheless, this approach is not suitable for the present application.

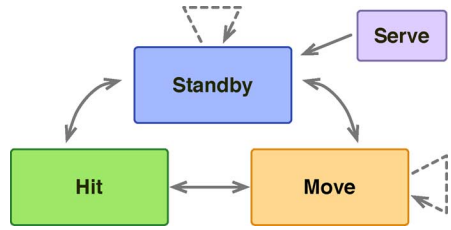

Fig. 6. Proposed four-state-transition model for tennis player behaviors.

In our opinion, a model that simulates the behavior of tennis players is needed. Therefore, we propose the four-state-transition model shown in Fig. 6. The four states are *Serve*, *Standby*, *Move*, and *Hit*. The arrows in the figure represent the allowable state transitions. All the behaviors of tennis players during a match can be expressed by these state transitions. For example, a player may move right, wait to hit, and then perform a forehand stroke—a sequence which can be modeled by state transitions from *Move*, to *Standby*, to *Hit*. In the case of serving, a player may serve, run to a standby position, wait for the opponent's return, run to a hitting position, and then hit—a sequence which can be modeled as *Serve*, *Move*, *Standby*, *Move*, and *Hit*. As in the intra-state transitions for *Standby* and *Move* shown by the dotted lines in Fig. 6, the same state may recur several times in rendering, the associated details of which are described in Section IV-A. Under this behavior model, the player database in Section III-A is further classified into four categories. As noted in Section II-D, four different hitting postures—forehand stroke, forehand volley, backhand stroke, and backhand volley—are collected in the *Hit* category. Subsequently, actions between two hits are classified in the *Move* category. Action clips without movement in the *Move* category are then collected to form the *Standby* category.

C. Database Reduction

Lots of motion clips with various running distances and directions are gathered in *Move* category, and many hitting postures are collected in *Hit* category. Nevertheless, most postures and movements in the database are similar because the player may perform the same posture several times in a three-hour match. The huge size of database would result in lagged rendering time due to searching the suitable successive clip in Section IV-A. To reduce the computation time in player rendering, a well-organized database is needed. Here the well-organized database

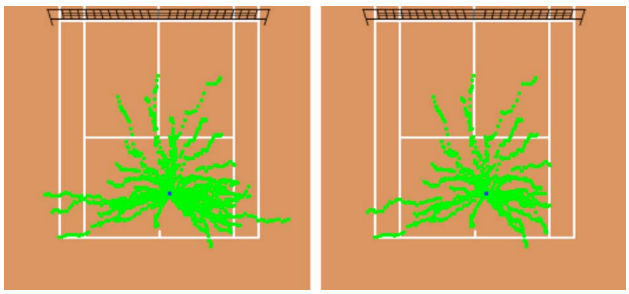

Fig. 7. Database reduction of *Move* category. There are 56 video clips in the original database (left part), and there are 30 video clips in the reduced database (right part).

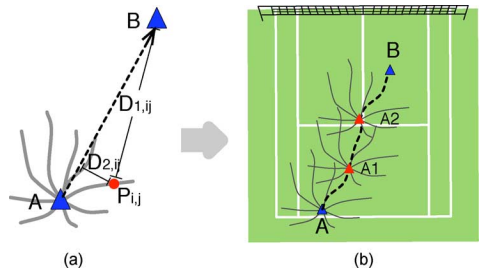

Fig. 8. (a) Selecting a suitable motion clip to form a movement path from position A to position B by computing similarity. (b) Three motion clips forming a movement path. (a) Factors of similarity computation. (b) Selection result from A to B.

stands for rendering various player postures with minor data size.

For database reduction in *Move* category, the dispersion and length of motion paths are considered as the factors. We see that more vivid moving postures can be rendered if motion clips in database have much various running directions and longer moving distances. In other words, we should keep the variety of motion paths in database and reduce their redundancy. The trajectory of each motion clip can be recorded as a moving vector with vector angle θ_i and vector length l_i. Here, 360 degrees of moving direction are segmented into small buckets, and each bucket has 30 degrees. Motion clips are classified into different buckets according to their vector angles. For example, the clip is gathered into bucket s if the vector angle θ_i is in the range $30(s-1)^\circ \leq \theta_i < 30s^\circ, \forall\ s \in [1, 12]$. It supposes that there are m clips in bucket s, $\mathbf{P_s^m}$, and we try to preserve only n paths in bucket s, $\mathbf{P_s^n}$, which have the maximum value of dispersion. The computation of dispersion is in direct proportion to variance of vector angles and moving distances, and the equation is written in the equation:

$$Dispersion(\mathbf{P_s^n}) = \frac{\sum_{i=1}^{n} (\theta_i - \bar{\theta_s^n}) l_i}{n} \tag{6}$$

where $\mathbf{P_s^n} \subseteq \mathbf{P_s^m}$ and $\bar{\theta_s^n}$ is the average angle of n selected paths in bucket s. For example, the maximum quantity of motion clips is 36 in the *Move* category if n is set to 3. As the example of database reduction in Fig. 7, the original database in the left part contains 56 motion paths, and the reduced database only has 30 motion paths. The dispersion after database reduction is similar to original database because 26 discarding paths are almost redundant to the others. For database in *Hit*, *Standby*, and *Serve* categories, there are some redundant gestures. In the experiments, we find that it is difficult to define an evaluative function to preserve the principle hitting/standby/serving postures. Therefore, we randomly select some clips in each subcategory to form the reduced database.

IV. Player Rendering

In this section, the methods of player selection are proposed in which suitable clips are selected from the database to render the various motions and postures of players. For seamless connection between clips, the proposed approaches can smooth transitions in the player's shape, color, and motion. Subsequently, the game system is integrated with the rendered background court and foreground objects.

A. Database Selection

In previous studies focusing on video-based rendering, Schodl *et al.* [5] and Schodl and Essa [16] extracted textures of a video and synthesized new video clips by rearranging frames in the original video sequences. In both studies, the measure of similarity for video synthesis is the pixel difference between two frames. However, the methods in these studies were not suitable for human rendering because the complexity of human images is higher than that of natural scenes used in previous studies. In previous studies of video-based rendering in humans, Kitamura *et al.* [20] used greater numbers of similarity measures such as motion vectors and contours, and both studies achieved more vivid visual effects. These studies should encourage the consideration of more similarity measures for better visual effects in human rendering. Nevertheless, the challenges are greater in player rendering because of the dedicated human videos different from users captured in [19] and [20], the player database is constructed from videos of matches. In other words, the database may be incomplete for some postures, and thus lead to more difficult rendering. Therefore, in the proposed method for player rendering, the first step is to select appropriate player clips.

1) Clip Selection for Player Moving: Suppose a motion path is to be rendered from position A to position B as illustrated in Fig. 8(a), and that the best path is the shortest one (dotted line). Occasionally, it is difficult to find a clip from the player database that fits the dotted line, and multiple moving clips must be connected to form the moving posture. A critical problem is how to choose suitable moving clips to achieve seamless connections. Suppose there are n moving clips, and moving clip i, $i \in [1, n]$, is composed of l_i frames. $P_{i,j}$ is the player position in frame j of moving clip i in real world coordinates, $j \in [1, l_i]$. All player positions can form the possible movement trajectories such as those shown by the grey lines in Fig. 8(a). The process of moving clip selection is to choose a motion path $[P_{i,1}, \ldots, P_{i,j}]$ which connects positions A and B. Three primary factors should

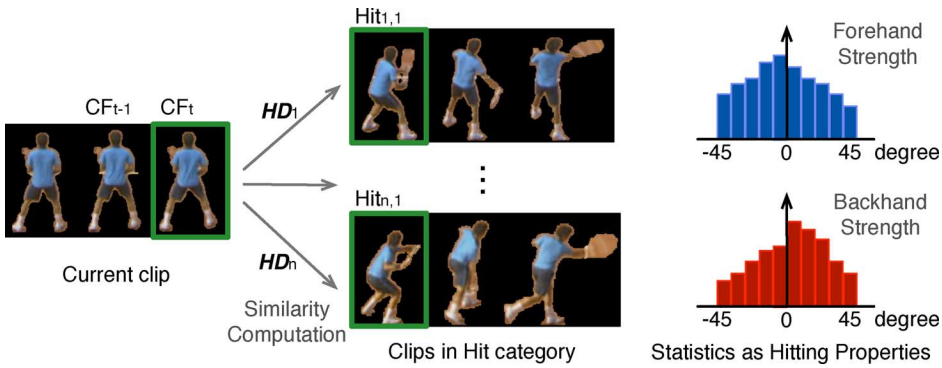

Fig. 9. Suitable clip is chosen when the hitting clip has higher similarity to the current clip. The hitting properties of a player depend on statistics from real videos. The blue and red charts show hitting statistics for forehand and backhand strengths, respectively, in each direction.

be considered in the selection of moving clips: distance $\mathbf{D_{1,ij}}$, distance $\mathbf{D_{2,ij}}$, and index j.

- $\mathbf{D_{1,ij}}$ is the distance from position $P_{i,j}$ to position B. The selected moving clip should shorten the distance to the destination position B. As shown in Fig. 8(a), the shorter the distance $\mathbf{D_{1,ij}}$, the better the motion path $[P_{i,1}, \ldots, P_{i,j}]$.
- $\mathbf{D_{2,ij}}$ is the distance from position $P_{i,j}$ to the shortest path. The selected moving clip should not be far from the shortest path. A shorter $\mathbf{D_{2,ij}}$ indicates that the rendered motion path from the selected clip is more efficient for player movement.
- A larger j is preferred because longer clips would make the rendering result smoother. In other words, a smaller j indicates that many shorter clips may be needed to render the motion path. This makes the path less smooth.

Taking these factors into consideration, the decision function at each position $P_{i,j}$ in moving clip selection is formulated as the sum of $\mathbf{D_{1,ij}}$, $\mathbf{D_{2,ij}}$, and j:

$$MD_{ij} = D_{1,ij} + D_{2,ij} - c_m j \tag{7}$$

where c_m is a weighting coefficient to balance the effectiveness of clip length. The moving clip with the minimum value of MD_{ij} is recognized as the most suitable. Occasionally, the selection procedure is repeated several times and multiple clips are selected to form the motion path. As the illustration in Fig. 8(b) shows, three clips, $\overline{AA_1}$, $\overline{A_1A_2}$, and $\overline{A_2B}$, form the motion path. It should be noted that the time to move from position A to position B depends on the player database. In other words, a shorter running time is needed if the player moves fast in various directions in the real video, which is recorded in the database as shown in Fig. 2(d). The rendering of player motion based on extracted clips containing the movement characteristics of the player in videos of real match is one of the distinguishing features of TRP.

2) Clip Selection for Hitting: Unlike *Move* clip selection, the major factor considered in *Hit* clip selection is the similarity of texture and shape between the final selected moving clip and the hitting clip. As the transition example in Fig. 9 illustrates, the primary challenge is how to choose the hitting clip $Hit_{i,1}$ to connect to the current frame CF_t in a seamless cascade. A reasonable assumption for a suitable connection is that the initial frames in a successive clip should be visually similar to the current clip. This requires computing the similarity between frames of the current clip and the initial frames of clips in the *Hit* category. HD_i, the distance between the initial frame of clip i and the current frame, is defined on the basis of textural and shape features. It is written as

$$HD_i = D_{tex,i} + c_h D_{shape,i} \tag{8}$$

$$D_{tex,i} = \sum_x \sum_y |Hit_{i,1}(x,y) - CF_t(x,y)|^2 \tag{9}$$

where $D_{tex,i}$ is the textural similarity between the current frame $CF_t(x,y)$ and the initial frame of the successive clip $Hit_{i,1}(x,y)$. Note that the positions of the player masks in these clips are normalized (details in Section III-A), and no alignment process is required in measuring textural similarity. $D_{shape,i}$ stands for the shape similarity derived with the Hausdorff-distance [21]. c_h is a weighting coefficient to balance the effectiveness of shape similarity. The clip Hit_i with the minimum HD_i is chosen as the successor clip.

For hitting properties, the hitting strengths are based on statistics from the player's performance in the match videos. As an example in Fig. 9, the blue and red charts show statistics for forehand and backhand strengths, respectively, in each direction (extracted in Section II-D). To simulate the game, a Gaussian variable is added to the direction and strength of each hit. Note that different players in different videos have different statistics—a property which makes the proposed interactive game system more realistic. The rendering of hitting based on player characters is another distinguishing feature of TRP.

B. Smoothing Transitions

Occasionally, the next selected clip may not be sufficiently similar to the current clip, in which case the rendering result will appear awkward when the two clips are directly connected. For an example of dissimilar clips in Fig. 9, the rendering result is not smooth if we cascade CF_t and $Hit_{1,1}$. In our observations, the dissimilarity between two clips comes from shape, color, and motion. To smooth the transition, we propose to insert transition frames between two cascading clips. The transition frames are calculated from the current clip and the next selected clip by considering the smoothness of shape, color, and motion. The

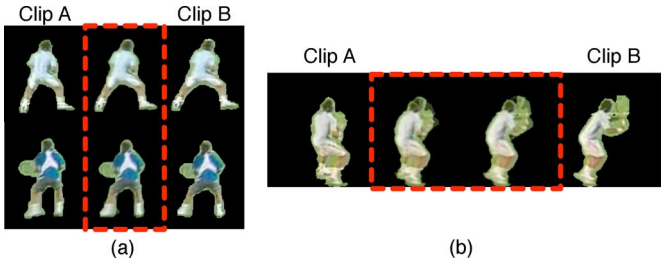

Fig. 10. Shape smoothing. (a) Insert one transition frame. (b) Insert two transition frames.

number of transition frames can be dynamically determined by the value of the similarity measure in (8). The higher value of HD_i, the larger number of transition frames insert.

For shape smoothing, we attempt to interpolate the transition postures between two cascading clips. A well-known approach for shape transition is the image morphing method proposed by Seiz and Dyer [22]. With transition points labeled by users, image morphing can generate smooth transitions between two different images. Furthermore, a hierarchical and tile-based image warping scheme proposed by Gao and Sederberg [23] can improve the results. However, neither method can be directly applied to our application because it is impossible to manually label the transition points between any two clips in a massive database. Therefore, another challenge is how to automatically label the transition points. With the help of the feature detection method proposed in [24] and the feature descriptor in [25], feature points on the images can be automatically detected and matched. Suppose we wish to find the transition frames between two images: $I_1(i,j)$ and $I_2(i,j)$. $P_1(k)$ and $P_2(k)$ are the positions of feature points on $I_1(i,j)$ and $I_2(i,j)$, respectively. We propose to modify the cost function for view morphing by adding the distance between feature points as shown in the following equation:

$$W = \sum_i \sum_j |I_1(i,j) - \tilde{I}_2(i,j)|^2 + \lambda \sum_{k=1}^{n} |P_1(k) - \tilde{P}_2(k)|^2 \tag{10}$$

where $\tilde{I}_2(i,j)$ is the warping result of $I_2(i,j)$, $\tilde{P}_2(k)$ are the positions of the feature points in $\tilde{I}_2(i,j)$, n is the number of matched feature points, and λ is the weighting coefficient. The morphing process employs hierarchical and tile-based warping by minimizing the cost function (10). The process iteratively finds the minimum value of W and stops when W converges. As an example of shape smoothing in Fig. 10(a), we insert one transition frame between clips A and B in which the player in the transition frame has an intermediate posture. Fig. 10(b) is another example of shape smoothing with two transition frames. It can be observed that transition frames can effectively smooth the clip connection.

For color smoothing, the clips in the database are segmented from different time periods in a video. Therefore, each clip may have a different background because of changes in the weather. Occasionally, changes in background lighting conditions lead to luminance variation in the clips, making the transition unpleasing. To solve this problem, all clips in the database are normalized to the color of the court with Poisson Editing [26].

For motion smoothing, clips may have different movement speeds and directions. A discontinuity in the motion will then make the rendering result discrete. To reduce discontinuities in motion, we provide an intermediate motion state to the transition frames. The intermediate motion state can be a linear interpolation of speed and direction between the two clips.

V. Game System

A. 3-D Scene Modeling and Rendering

The entire game system includes not only the player rendering but also the rendering of background scene. The background scene is also an important component of the game system because a better rendering of it will increase the game's reality and user's enjoyment. Inspired by the method proposed by Horry *et al.* [27] in "Tour Into the Picture" and the improved methods in [28], in TRP, 3-D scenes are rendered from a 2-D image once the user manually labels the 3-D structure of the image. As the illustration in Fig. 11 shows, the 3-D structure of a tennis court can be roughly modeled by seven boards: 1) the floor, 2) the base of the rear audience, 3) the top of the rear audience, 4) the base of the left audience, 5) the top of the left audience, 6) the base of the right audience, and 7) the top of the right audience. It is difficult to automatically extract the 3-D information of the court model because the court layout is different from various matches. For previous works, Han *et al.* [29] proposed methods to automatically calibrate the court-net model but it still cannot extend to the scene calibration. Here, all the 3-D information of boards of the court scene is manually labeled, and the demo video in the paper shows an example of manually labeling. The 2-D scene rendered from the 3-D structure is controlled by intrinsic and extrinsic parameters of the camera as follows:

$$\begin{bmatrix} hx \\ hy \\ h \\ 1 \end{bmatrix} \sim \begin{bmatrix} f_0 & 0 & x_0 & 0 \\ 0 & f_0 & y_0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} [\mathbf{R} \mid \mathbf{t}] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$[\mathbf{R} \mid \mathbf{t}] = \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_x \\ R_{21} & R_{22} & R_{23} & T_y \\ R_{31} & R_{32} & R_{33} & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{11}$$

where f_0 is focal length and $[x_0, y_0]$ are the offset coordinates of the intrinsic parameters. The rotation matrix $\mathbf{R}$ and translation matrix $\mathbf{t}$ are extrinsic parameters. By modifying these camera parameters, we can render virtual 2-D scenes from the 3-D structure in any viewing angle. For instance, the camera panning or tilting can be modeled with modifying parameters in $\mathbf{R}$, and a close look of the player can be implemented by increasing the focal length f_0. Incorporating foreground rendering into the 3-D structure, the foreground objects in Fig. 11 are rendered in the following order: Player B, net, referee, ball boy, ball,[6] and Player A. In order to achieve more vivid player effects, we also model the light source and draw player shadows by warping player shapes according to the position of light sources. The light conditions of players and tennis courts in the game may be

[6]If the depth of the ball is deeper than that of the net, the rendering priority of the ball is higher than that of the net.

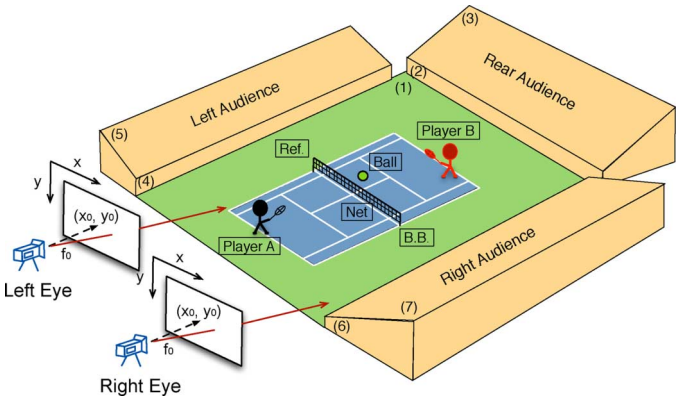

Fig. 11. Game system combines the 3-D model and video-based rendering. The background is modeled as seven boards and the rendering order is labeled by the indices. The foreground objects are rendered in their order of depth: Player B, ball, net, referee, ball boy, and player A. 3-D viewing function is implemented in our system by rendering right-eye and left-eye views in the same time.

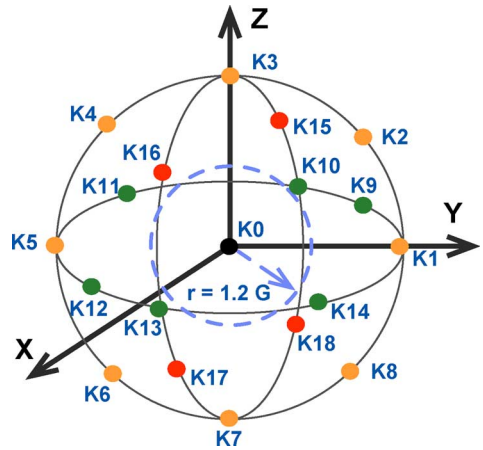

Fig. 12. The XYZ-axis acceleration signals of Wiimote are quantized to 19 features. A signal is classified to K0, if the signals magnitude is less than 1.2 times of gravity.

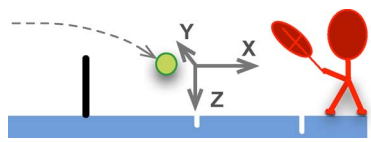

Fig. 13. Ball in the tennis games is forced by: player's hit, gravity, air friction, and court friction. There needs to be a precise model to model the ball trajectory in the game.

different when the players and game court belong to different match videos. For example, the players in Wimbledon Open compete on the court of French Open, and game frame would look weird without any processing. Thus, we adjust the color tone of players to fit that of game court by Poisson Editing [26]. To model the motion blur effect of a fast-moving ball, alpha blending and multiple-ball rendering are used. More vivid rendering results can be generated by attending to these details.

3-D visual effect becomes popular in recent years, and more consumers purchase the television with 3-D viewing functions. However, the shortage of 3-D video contents is still a problem that decreases the functionality of 3-D TV. Since the models of background scenes and foreground objects are constructed on the 3-D world, and the 3-D visual effect of game frames can be implemented by rendering right-eye and left-eye views in the same time. Fig. 11 is the illustration, which shows that the right-eye and left-eye views are rendered with a separation of eye-distance.

B. Signal Analysis of Wiimote

The user dialogue is a key component to increase the interactivity in our system. From the great success of Wii Sports, we know that the interface of user dialogue can increase more interaction and bring more fun during the game. Therefore, we try to analyze the signals from Wiimote and transfer these signals into corresponding hitting gestures. Before signal analysis, we should find a way to receive the signals of XYZ-axis acceleration from Wiimote. There are some source codes on the Internet, like GlovePIE [30] or WiiYourSelf [31], to detect the Wiimote device and receive these signals through Bluetooth protocol. For the research studies on signal analysis of Wiimote, Schlomer *et al.* [32] used hidden Markov model (HMM) to analyze the acceleration signals of XYZ-axis for gesture recognition. We adapt Schlomer's methods to recognize tennis gestures: forehand stroke, forehand volley, backhand stroke, backhand volley, and serve. The experimental results show that the average accurate rates are 85%. To further improve the recognition rates, we replace the HMM by support vector machine (SVM) as the feature classifier. First, we collect the acceleration signals from Wiimote with the sampling rates: 30 samples per second, and then these signals are quantized into 19 feature vectors as shown in Fig. 12. To prepare the feature vectors for SVM, we transform temporal signals into spatial signals by calculating the histogram of quantized signals during a half second. Finally, a 19-bin histogram is the feature vector for the SVM classifier, and we employ libsvm [33] to implement SVM classifier in the experiments. In the results, the average recognition rates, including forehand stroke, forehand volley, backhand stroke, backhand volley, and serve, are improved to 95%.

C. Moving Model of Ball

The model of ball trajectories and moving states are also significant in a game system because the users would pay lots attention on the ball in playing games. We find that a ball in the tennis games is forced by: player's hit, gravity, air friction, and court friction. Therefore, a moving model with considering the four factors is proposed to render a vivid ball trajectory. We extend the concepts of Kalman-based filter, which is an efficient recursive filter that estimates the state of a dynamic system from a series of incomplete and noise measurements [14], to construct the moving model. The ball trajectory can be modeled as follows:

$$\mathbf{x_t} = \mathbf{F_t}\mathbf{x_{t-1}} + \mathbf{H_t}\mathbf{u_t} + \mathbf{C_t}\mathbf{u_t} + \mathbf{w_t} \qquad (12)$$

where $\mathbf{x_t}$ models the state of the ball position, velocity, and acceleration at time t as shown in Fig. 13, $\mathbf{u_t}$ is the unit vector of ball state, and $\mathbf{w_t}$ is the process noise. Because the gravity and air friction are the force exerted on the ball in the air, the state transition $\mathbf{F_t}$ can be modeled as a second-order motion. The control-input $\mathbf{H_t}$ can be used to model the ball hit by

TABLE I
INFORMATION OF PLAYER DATABASE AND THE MATCH VIDEOS WHERE THE PLAYERS EXTRACTED FROM

Tennis Video	**Player's Name**	**Clip Number in** *Move*	**Clip Number in** *Hit*
2009 French Open Semi Final	Roger Federer	27	12
2009 Wimbledon Open Semi Final	Serena Williams	22	12
2007 Australia Open Final	Roger Federer	29	26
2009 French Open Semi Final	Juan Martin del Potro	28	12
2009 Wimbledon Open Semi Final	Elena Dementieva	22	12

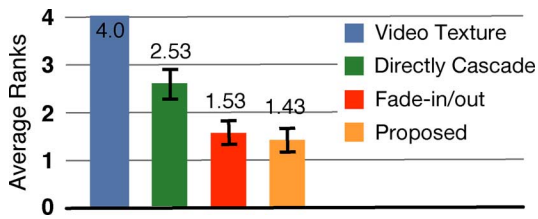

Fig. 14. Subjective evaluation for cascading methods in player rendering. Statistics of the ranks among different cascading methods.

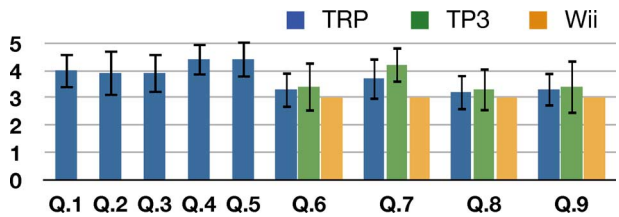

Fig. 15. Results of subjective evaluation. The bar heights are the average scores, and the black lines show the standard deviations.

players, and $\mathbf{C_t}$ models the court friction exerted on the ball when bouncing on the court. Therefore, (12) is rewritten as the following:

$$\begin{bmatrix} P_t \\ V_t \\ A_t \end{bmatrix} = \begin{bmatrix} F_p & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & F_v & \Delta t \\ 0 & 0 & F_a \end{bmatrix} \begin{bmatrix} P_{t-1} \\ V_{t-1} \\ A_{t-1} \end{bmatrix} + \begin{bmatrix} H_p \\ H_v \\ H_a \end{bmatrix} + \begin{bmatrix} C_p \\ C_v \\ C_a \end{bmatrix} + \begin{bmatrix} \omega_p \\ \omega_v \\ \omega_a \end{bmatrix} \quad (13)$$

where $[P_t \ V_t \ A_t]^t$ are the prediction results of state $\mathbf{x_t}$, $[P_{t-1} \ V_{t-1} \ A_{t-1}]^t$ are the state in time index $t-1$, and $[\omega_p \ \omega_v \ \omega_a]^t$ are the process noise. The values of $\mathbf{H_t}$ are extracted from real videos and present the player's performance in the match. We suppose the hitting strength from statistics is $\mathbf{H_s}$, the user's force feedback is $\mathbf{H_u}$, and the hitting strength, $\mathbf{H_t}$, of the player in the game is $\mathbf{H_s} \cdot \alpha\mathbf{H_u}$, where α is the parameter for the normalization. In other words, each player would have the unique $\mathbf{H_t}$ in the game, and this is also one of the distinctive features in the system. Different court has the different $\mathbf{C_t}$ in the game. For example, the court friction of French Open would be higher than that of US Open due to the higher friction of clay court than that of hard court. The ball trajectory would become more realistic with considering these real properties.

VI. EXPERIMENTS AND DISCUSSION

We designed a graphic user interface to show rendering results at a resolution of 720×480 [Fig. 2(a)], XY positions of players and ball in real world coordinates [Fig. 2(b)], YZ positions of players and ball in real world coordinates [Fig. 2(c)], and motion paths of players A and B [Fig. 2(d)]. In particular, Fig. 2(b) and (c) clearly illustrates that the game system is built upon a 3-D model. Fig. 2(d) shows the potential motion paths of players, all of which are based on a database extracted from real videos of matches to record movement properties. Table I shows the information of 5 tennis players in the game and also includes the match videos where the players extracted from. Because many *Move* and *Hit* clips in the player database are similar, we only select some clips based on the proposed criteria in Section III-C.

A. Results of Game Rendering

Further rendering results are shown in Fig. 17. Fig. 17(a) shows players competing on a court at the French Open. The segmented players from match video are usually shadow-removed. To give more vivid visual effects, shadows of foreground objects are added to the court surface that can give the rendering player with a steady stand on the court. The shadow is rendered by the player's shape with a tilt according to the lighting source. Furthermore, the score is seamlessly painted on the court with alpha blending and it will update with game proceeding. To increase gaming excitement, the player's hitting energy is shown accumulating during the game with the bars in the upper-right and lower-left corners. The player produces a powerful stroke accompanied by a fire ball when the hitting energy is full, as shown in Fig. 17(g) and (j). In addition, the player's body size will enlarge double when preparing to hit a powerful stoke that can bring users more excitement in playing games. Note that the hitting energy rapidly increases if the user gives a strong hitting strength. Fig. 17(b) shows players competing on a court at Wimbledon, and it demonstrates that the rendering effects are quite realistic and resemble real videos. In order to provide various game difficulty, we created some animated characters, like the boss in a game, who have different hitting properties and give users more fun during the game. Fig. 17(c) shows two animated characters on a court at the US Open. The position of the camera can be changed, for example the rendering scene behind player A in Fig. 17(d)–(f). With changes in the viewing angle, the visual effects of TRP are more vivid and offer more novel experiences to users. A demo video showing the player and background renderings is available from the link of system overview in Section I.

(a) (b) (c)

Fig. 16. Demonstration in public. Thousands of people have experienced the innovation of TRP. (a) 2010 Taipei International Invention Show & Technomart. (b) 2010 Taichung Information Technology Month. (c) 2010 Kaohsiung Information Technology Month.

(a) (b) (c)
(d) (e) (f)
(g) (h) (i)
(j) (k) (l)

Fig. 17. Rendering results. (a),(d) Two players competing on a French Open court with different viewing angles. (b),(e) Two players competing on a Wimbledon court with different viewing angles. (c),(f) Two animated characters competing on a US Open court with different viewing angles. (g),(j) The player performs a powerful strike when energy bar is full. (h),(k) Left-eye and right-eye views of rendering results, respectively. (i),(l) Player's color tones are modified by Poisson Editing.

As mentioned in Section V-A, TRP can support 3-D viewing function of game frames by modifying rotation matrix $\mathbf{R}$ and translation matrix $\mathbf{t}$. Fig. 17(h) and (k) shows the rendering results of left-eye and right-eye views of game frames, respectively, and we could find that the corresponding distances between the player and score on the court are different from right-eye and left-eye views. The function of 3-D visual effects gives users more immersive experiences in playing games. The light conditions of players and tennis courts in a game may be different when the players and game court belong to different match videos. Thus, we adjust the color tone of players to fit that of game court, and Fig. 17(i) and (l) is the results after the players processed by Poisson Editing [26]. For example, the original color tones of players in French Open are blended with brown color due to the clay court. That would cause a weird visual effect due to the different color tone, if we directly past

TABLE II
RESULTS OF HITTING MOMENT DETECTION IN TENNIS VIDEOS

Tennis Video	Quantity	Precision (%)	Recall (%)
2007 Australia Open Women's Single	153	97.7	85.0
2007 Australia Open Men's Single	145	96.9	86.2
2008 US Open Women's Single	148	98.5	87.8
2009 French Open Men's Single	160	96.8	75.6
2009 Wimbledon Open Women's Single	151	96.2	82.8
Total	757	97.2	83.4

the players in the court of US Open or Wimbledon Open. The Poisson Editing can make players' color tone blend with blue color in the court of US Open and green color in the court of Wimbledon Open. The process to change the color tones can make the players playing on the court more seamless. However, Poisson Editing is time-consuming and should be processed at each frame because the player's movement or viewing angle change would cause different boundary condition in the optimization process. To make the game system running with real-time requirement, we just set the constant boundary condition of the modification of player's tone but that make the rendering effect not seamless enough. How to rendering a seamless view in real-time under the matching count different from the gaming count is still a challenge.

To control the player in the game, we analyzed the XYZ-axis acceleration signals of Wiimote through the Bluetooth protocol for user's gesture recognition, as mentioned in Section V-B. The hitting strength of a player in the game is proportional to the statistics in the match video and user's force feedback. We suppose the hitting strength from statistics is $\mathbf{H_s}$, the user's force feedback is $\mathbf{H_u}$, and the hitting strength of the player in the game is $\mathbf{H_s} \cdot \alpha\mathbf{H_u}$, where α is the parameter for the normalization. The moving direction of ball in the game is also controlled by the hitting timing. The earlier hit makes the ball deviate left, whereas the later hit will make it deviate right. For game audio, the hitting sound is extracted from the match video when collecting the clip database. Each clip in Table I contains both video and audio information. The match audio would play during the game synchronic with the player's posture. We also collect the audiences' cheer and dynamically play it when the player performs a nice rally.

B. Evaluation of Hitting Detection

For the method of hitting detection in Section II-C, we set a second-order model in the experiments, and the number of non-existing candidate is set to 1 in the trajectory complement. Five different tennis videos in Table II are used as test sequences for the detection of hitting moment. Due to the thousands of hitting times in a match video, we randomly pick some for the statistics and manually label hitting times to be target values. To determine the detected hitting time is correct or not, we set a decision rule that the time difference between detected result and target value less than 0.2 s would be considered as correct. If the time difference was large than 0.2 s, then it would be considered as false. Table II shows that the average precision rate is 97.2%, and average recall rate is 83.4%. The high precision rate shows the proposed method can accurately detect the hitting time. With higher accuracy of hitting time detection, we can also get higher accurate statistics of hitting strengths and directions. In addition, the decrease in recall rate was caused by the loss of ball detection in a period of time in foreground segmentation. We can set a large number of non-existing candidate to improve the loss detection in foreground segmentation, but it sometimes would make the precision rate of trajectory extraction decreased.

C. Evaluation for Clip Transition

As mentioned in Section IV-B, we propose a cascading method to smooth the clip transition, which detects the feature points on clips, finds matching pairs between adjacent clips and employs the modified morphing (10) to calculate the transition frames. From the experimental results, the proposed method can accurately find the matching pairs and calculate transition frames correctly in most cases, but there are still some rendering clips not smoothing enough. We think there are two factors making the proposed method fail. One is a few number of matching pairs, and the other is the non-convergent result in (10) due to the short computation time by real-time requirement. A few number of marching pairs may lead the latter part of (10) omitted that would make the rendering results just converge to local minimum. Because the morphing process needs to iteratively refine the transformation, the much dissimilar clips will need more computation time to get better results. However, the requirement of real-time performance is always the critical factor in the game rendering; thus some clip connections cannot be refined to be perfect during the game. The number of discontinuous motion will decrease if the computer as the game server has higher computation capability.

To compare the performance of clip transition, four methods, Video Texture[5], Cascade Directly, Fade-in/out, and the proposed method, are implemented. To justify the visual effect of smoothing transition, a survey was designed and 32 subjects, who are graduated students and never saw this work before, were asked to rank these video clips rendered by the four methods, where ranking 1 means the most natural rendering and ranking 4 means the most unnatural rendering. As Fig. 14 presents, the proposed method has the average rank 1.43 with standard deviation 0.65, which is the highest rank among these methods. It should be noticed that the visual effect of fade-in/out is higher than the directly cascade and even similar to our proposed method. We think one of the reasons may be the few number of insertion frames to attract user's attention.

TABLE III
COMPUTATION DETAILS PER FRAME

Item	Computation Units
Clip Selection	45
Smooth Transition	$160 \sim 270$
Background Rendering	100
Foreground Rendering	$20 \sim 90$
Others	40

From the human perceptual report [34], people cannot notice the tiny difference when the change happening within 0.1 s. In other words, the number of insertion frame must be larger than 3 in a 30 fps rendering system to attract user's attention. In our opinion, fade-in/out method can be a candidate as cascading method when the computational ability is insufficient in a real-time system. However, we think the proposed method can achieve a better ranking if a larger resolution of video clips is available to provide more feature points and matching pairs. A powerful game server can also render more smoothing results within the short rendering time by real-time requirement. There is an interesting result that the visual effect of Video Texture is the lowest and even worse than the directly cascade. Because the human behavior is much more complex than their experiments [5], and the cost function should be modified with considering more items.

D. Computational Analysis

TRP is an interactive tennis game and requires real-time rendering performance during user interaction. The computation for game rendering includes clip selection, smoothing transitions, background rendering, and foreground rendering. We set the computation of background rendering as 100 computation units (CUs) per frame and normalized the CUs for the other steps as shown in Table III. Foreground rendering requires 20 to 90 CUs per frame depending on the position of camera. For example, the computational load is heavy when the camera position is close to the player such as the Fig. 17(e) because a larger foreground area have to render. Note that the computations for background rendering do not decrease when foreground computations increase because the background rendering is independent of the foreground rendering in our rendering scheme.

Clip selection would process and depend on the current pose of the user. Due to the partial selection of player database, clip selection only costs 45 CUs, and it would be linearly increased when clip number increases. For example, the CU of clip selection would increase 1.87 times if the clip number is 56 but not 30 in Fig. 7. Smoothing transitions require extensive computations to detect feature points, calculate matching pairs, and execute the morphing process. In the experiments, smoothing transition requires 160 to 270 CUs per frame depending on the size of foreground players, which costs the most computation in the game rendering. The others are the computation of signal analysis of Wiimote, computation of ball trajectory, and so on. We made a multi-thread program and employed a PC with Intel i7 2.6-GHz CPU to achieve a rendering performance of 720×480 and 30 fps, providing users with a more comfortable gaming experience.

E. Game Prediction With Player Statistics

Player rendering with hitting statistics and movement properties extracted from real videos is a key feature of TRP. A player's performance in TRP may reflect that player's performance in a real video. Therefore, we designed an experiment to observe whether the performances of players in TRP correlate with those in a real video. The experimental results are shown in Table IV. Two real videos were used: the men's semi-final of the French Open and the women's semi-final of Wimbledon, both in 2009. From the match records, the percentage of games won by Roger Federer in the former is 51%. The percentage of games won by Serena Williams in the latter is 54%.

To simulate a match with TRP, both players were controlled by the computer. A Gaussian variable was added in the direction and strength of hits to model the player in the real video. The simulations were run for 5 and 3 sets for the French Open and Wimbledon, respectively. The percentage of games won by Roger Federer was 55%, and that by Serena Williams was 61%. Therefore, the performance of a player in TRP seems to reflect that player's performance in the real videos, although the former slightly overestimates the latter. We realize that match results are difficult to predict because player's performance depends not only on hitting and moving but also on emotions, the weather, and chance. Nevertheless, we might still test how well the simulated results of TRP hold in general. For example, it would be interesting to use TRP to predict the results of Federer's performance in the 2009 French Open final and in the 2007 French Open. This could potentially show whether Federer's technique has advanced or regressed.

F. Subjective Evaluation of Game Play

For the user study, we designed subjective evaluations for 20 undergraduates who played TRP for the first time, and the game environment was captured in our demo video. Of the 20 evaluators, 11 had a habit of watching videos of tennis matches whereas the rest did not. Sixteen evaluators had a habit of playing games on PS3 or Wii, whereas the others did not. Five questions were designed to evaluate the experience of playing TRP, and four were designed to compare the experiences of playing TRP, Wii Sports, and Top Spin 3 on PS3.

To evaluate if the viewers could have any experience from playing TRP after watching match videos, we have to ask subjects to watch match videos as the experimental condition. Subsequently, they were required to play TRP and score their satisfaction on a five-point scale, i.e., 1, very unsatisfied; 2, somewhat unsatisfied; 3, no difference; 4, somewhat satisfied; and 5, very satisfied. The five questions were as follows:

Q.1) Did you have interactions with the video content from playing TRP?

Q.2) Did you have an immersive experience with the game of tennis from playing TRP?

Q.3) Was it entertaining and interesting to play TRP?

Q.4) Do you think that TRP is an innovative multimedia application?

Q.5) Are you more willing to play TRP after watching videos of tennis matches?

The average scores and standard deviations of the evaluations are listed in Table V, where Class A is the results of 11

TABLE IV
PERCENTAGE OF GAMES WON IN REAL VIDEOS AND IN RESULTS SIMULATED BY TENNIS REAL PLAY

Game Video	Name of Player A	Name of Player B	Game Points A-B	Game(%)	Simulation(%)
2009 French Open S.-F.	Roger Federer	Juan Martin del Potro	3-6, 7-6, 2-6, 6-1, 6-4	51 : 49	55 : 45
2009 Wimbledon Open S.-F.	Serena Williams	Elena Dementieva	6-7, 7-5, 8-6	54 : 46	61 : 39

TABLE V
AVERAGE SCORES AND STANDARD DEVIATIONS FOR **Q.1** TO **Q.5** OF USER STUDY

	Q.1		Q.2		Q.3		Q.4		Q.5	
	Avg.	SD	Avg.	SD	Avg.	SD	Avg.	SD	Avg.	SD
All evaluators (20)	4.0	0.67	3.9	0.85	3.9	0.74	4.4	0.60	4.4	0.68
Class A (11)	4.2	0.63	4.1	0.88	3.9	0.88	4.6	0.52	4.8	0.42
Class B (9)	3.8	0.67	3.8	0.83	3.9	0.60	4.1	0.60	3.9	0.60
Class C (16)	4.0	0.65	3.9	0.83	3.9	0.70	4.5	0.52	4.5	0.64
Class D (4)	4.0	0.82	4.3	0.96	3.8	0.96	4.0	0.82	4.0	0.82

TABLE VI
AVERAGE SCORES AND STANDARD DEVIATIONS FOR **Q.6** TO **Q.9** OF USER STUDY

	Q.6				Q.7				Q.8				Q.9			
	TP3		TRP		TP3		TRP		TP3		TRP		TP3		TRP	
	Avg.	SD	Avg.	SD	Avg.	SD	Avg.	SD	Avg.	SD	Avg.	SD	Avg.	SD	Avg.	SD
All evaluators (20)	3.4	0.91	3.3	0.67	4.2	0.65	3.7	0.78	3.3	0.80	3.2	0.66	3.4	1.00	3.3	0.62
Class A (11)	3.9	0.80	3.5	0.71	4.5	0.67	3.8	0.90	3.7	0.83	3.4	0.67	3.7	0.80	3.5	0.71
Class B (9)	3.1	1.00	3.1	0.63	4.0	0.53	3.8	0.61	3.0	0.84	3.0	0.58	3.1	1.18	3.2	0.45
Class C (16)	3.5	0.95	3.3	0.72	4.3	0.68	3.7	0.85	3.3	0.85	3.2	0.72	3.4	0.85	3.3	0.67
Class D (4)	3.3	0.89	3.2	0.52	4.0	0.52	3.7	0.50	3.3	0.68	3.1	0.38	3.3	1.61	3.4	0.46

evaluators with a habit in watching tennis matches, Class B is the results of 9 evaluators without a habit in watching tennis matches, Class C is the results of 16 evaluators with a habit in playing games, and Class D is the results of 4 evaluators without a habit in playing games. Fig. 15 illustrates the average results of subjective evaluation among all users. The results show that evaluators identified with increased interaction, immersive experience, and enjoyment from playing TRP. Furthermore, they highly agreed that TRP is an innovative multimedia application and are more willing to play it after watching videos of tennis matches. In addition, the evaluators with a habit in watching tennis matches gave higher score than the others. It seems that evaluators with watching tennis matches had higher satisfaction and were more eager to demand the functions provided by TRP.

In the next phase, evaluators were required to play the tennis games in Wii Sports(Wii), Top Spin 3(TP3) on PS3, and TRP. They were told to use Wii as the standard of comparison and give a score of 1 to 5 for their experience with TP3 and TRP, i.e., 1, much worse; 2, somewhat worse; 3, no difference; 4, somewhat better; 5, much better. The four questions were as follows:

Q.6) Comparing the entertainment levels of each game, what do you think of the performance of TP3 and TRP?

Q.7) Comparing the realism of the visual effects, what do you think of the performance of TP3 and TRP?

Q.8) Comparing the interactiveness of each game, what do you think of the performance of TP3 and TRP?

Q.9) Comparing your preferences for each game, what do you think of the performance of TP3 and TRP?

The average scores and standard deviations of the evaluations are listed in Table VI. Note that the scores of Wii are all 3 and deviations are 0 because we take Wii as the comparison basis. The primary advantages of Wii are the innovations in user-interactive dialogue (e.g., Wiimote). TRP also employs Wiimote for interactive dialogue. From the results in Fig. 15, the performances of TRP in regard to visual effects, interaction, and preference are all higher than for Wii. Some evaluators noted that TRP has vivid rendering effects and realistic player properties which provided them with a more interesting and enhanced experience. Compared to TRP, the primary advantages of TP3 are its vivid rendering effects of the court and the players. The performances of TRP are slightly lower than those of TP3 in terms of entertainment, visual effects, and preference. However, we feel that the performances of TRP are still outstanding because TP3 requires dozens of individuals to build the game model and draw textures. In contrast, all of the materials in TRP are simply extracted from real videos. Furthermore, this feature may also lead to a new framework in game production; the latest game of TRP will be available after a real tennis match is played. It should also notice that the scores from evaluators with a habit in watching tennis matches are higher than that of the others. TRP seems hold a higher attraction for people with habits in watching tennis matches.

For the technology promotion, TRP were invited by several exhibitions including 2010 Taipei International Invention

Show & Technomart,[7] 2010 Taichung Information Technology Month,[8] and 2010 Kaohsiung Information Technology Month[9] to demonstrate in public, and the demonstration session in International Conference on Consumer Electronics 2011[35] along with Consumer Electronics Show. As Fig. 16 shows, thousands of people have enjoyed the innovation from playing TRP. Most users said that they never played a vivid video game like TRP, and they would like to have it in their living room. Some public even said they would like to buy a TV with providing TRP functions, and they could play the video game after watching the match video everyday. The video of live demonstration in Taipei International Invention Show & Technomart is available on our website.

VII. Conclusion and Extension

Inspired by video analysis/annotation, video-based rendering, and interactive sports games, an interactive tennis game—TRP—constructed using models extracted from videos of real tennis matches is proposed. As techniques for player model creation, we propose the process for database normalization and the 4-state-transition behavioral model of tennis players. For player rendering, we propose clip selection, smoothing transitions, and the framework combining a 3-D model with video-based rendering. Experiments show that vivid rendering results can be generated with low computational requirements. Moreover, the player model can adequately record the ability and condition of a player, which can then be used to roughly predict the results of real tennis matches. User studies reveal that subjects like the increased interaction, immersive experience, and enjoyment from playing TRP. They also show that evaluators rate the visual effects, interaction, and preference for TRP higher than those for Wii Sports but slightly lower than those for Top Spin 3. However, unlike building complex scene models or drawing player textures in Top Spin 3, all of the materials in TRP are extracted from videos of real matches. This property can also provide a new framework for game production; the latest game of TRP will be available after a tennis match is played.

For further improvements, some steps of the application can be improved by utilizing a powerful computation server discussed in Section VI-C or the existing techniques. For example, a few recent methods for shape [36], [37] and color interpolation [38] which might be useful to render more vivid viewing effects. However, the real-time constraint is one of reasons why we do not prefer these techniques. To accelerate the computation and achieve a better rendering result, we had developed a hardware chip[39] embedded in the TV system to render TRP for resolution 720×480 with 30 fps, and we are currently developing the extensions to support rendering specification for resolution 1920×1080 with 30 fps.

Limitations of current system include the restrictions in the viewing angles and resolutions of the rendered game frame. For example, the system cannot render arbitrary views of the player, and the rendered game frame is blurry if the resolution of the match video is insufficiently high. Nevertheless, the limitation in viewing angles can be overcome if multiple videos from different cameras are made available. By constructing a database of multiple court models and players, the system can render game frames from any viewing angle. To overcome the limitation of low resolution, super-resolution techniques may be employed to preserve more details from the real video.

For future studies, our top priority is to extend the application of the proposed methods to other sports videos. The proposed methods in player model creation and player rendering will be modified. For example, the techniques of database normalization, clip selection, and smoothing transitions can be applied to videos of football games. Specifically, the proposed four-state-transition model of tennis players can be replaced by a transition model for football players (i.e., shot-pass-stop-motion). In this way, the framework in TRP can be extended to other sports videos to create games such as Football Real Play and Baseball Real Play.

[7]The exhibition is held in Taipei World Trade Center, and the period is from September 9, 2010 to October 3, 2010.

[8]The exhibition is held in Taichung ShuiNan Airport Exhibition Center, and the period is from December 17, 2010 to December 22, 2010.

[9]The exhibition is held in Kaohsiung Tuntex Sky Tower; the period is from December 30, 2010 to January 4, 2011.

References

[1] J. Wang, C. Xu, E. Chng, K. Wah, and Q. Tian, "Automatic replay generation for soccer video broadcasting," in *Proc. 12th Annu. ACM Int. Conf. Multimedia (MULTIMEDIA '04)*, 2004, pp. 32–39.

[2] J. R. Wang and N. Parameswaran, "Analyzing tennis tactics from broadcasting tennis video clips," in *Proc. 11th Int. Multimedia Modelling Conf.*, 2005, pp. 102–106.

[3] G. Zhu, C. Xu, Q. Huang, W. Gao, and L. Xing, "Player action recognition in broadcast tennis video with applications to semantic analysis of sports game," in *Proc. 14th Annu. ACM Int. Conf. Multimedia*, 2006, pp. 431–440.

[4] J.-H. Lai and S.-Y. Chien, "Tennis video enrichment with content layer separation and real-time rendering in sprite plane," in *Proc. IEEE Int. Workshop Multimedia Signal Processing (MMSP 2008)*, 2008, pp. 672–675.

[5] A. Schodl, R. Szeliski, D. H. Salesin, and I. Essa, "Video textures," in *Proc. 27th Annu. Conf. Computer Graphics and Interactive Techniques*, 2000, pp. 489–498.

[6] A. A. Efros, A. C. Berg, G. Mori, and J. Malik, "Recognizing action at a distance," in *Proc. 9th IEEE Int. Conf. Computer Vision*, 2003, pp. 726–733.

[7] N. Inamoto and H. Saito, "Free viewpoint video synthesis and presentation of sporting events for mixed reality entertainment," in *Proc. 2004 ACM SIGCHI Int. Conf. Advances in Computer Entertainment Technology*, 2004, pp. 42–50.

[8] J.-H. Lai, C.-L. Chen, C.-C. Kao, and S.-Y. Chien, "Tennis video 2.0: A new presentation of sports videos with content separation and rendering," *J. Vis. Commun. Image Represent.*, vol. 22, no. 3, pp. 271–283, 2011.

[9] J.-H. Lai, C.-L. Chen, P.-C. Wu, C.-C. Kao, and S.-Y. Chien, "Tennis real play: An interactive tennis game with models from real videos," in *Proc. 19th Int. Conf. Multimedia*, 2011, pp. 483–492.

[10] J.-H. Lai, C.-C. Kao, and S.-Y. Chien, "Super-resolution sprite with foreground removal," in *Proc. IEEE Int. Conf. Multimedia and Expo.*, 2009, pp. 1306–1309.

[11] X. Yu, N. Jiang, L.-F. Cheong, H. W. Leong, and X. Yan, "Automatic camera calibration of broadcast tennis video with applications to 3d virtual content insertion and ball detection and tracking," *Comput. Vis. Image Understand.*, vol. 113, no. 5, pp. 643–652, 2009.

[12] C.-H. Chang, K.-Y. Hsieh, M.-C. Chiang, and J.-L. Wu, "Virtual spotlighted advertising for tennis videos," *J. Vis. Commun. Image Represent.*, vol. 21, no. 7, pp. 595–612, 2010.

[13] J. Han, D. Farin, and P. H. de With, "A mixed-reality system for broadcasting sports video to mobile devices," *IEEE Multimedia*, vol. 18, no. 2, pp. 72–84, 2011.

[14] R. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 1, pp. 35–45, 1961.

[15] M.-C. Roh, B. Christmas, J. Kittler, and S.-W. Lee, "Gesture spotting for low-resolution sports video annotation," *Pattern Recognit.*, vol. 41, no. 3, pp. 1124–1137.

[16] A. Schodl and I. A. Essa, "Controlled animation of video sprites," in *Proc. 2002 ACM SIGGRAPH/Eurographics Symp. Computer Animation*, 2002, pp. 121–127.

[17] G. Colqui, M. Tomita, T. Hattori, and Y. Chigusa, "New video synthesis based on flocking behavior simulation," in *Proc. Int. Symp. Communications, Control and Signal Processing*, 2008, pp. 936–941.

[18] P. M. Phillips and G. Watson, "Generalising video textures," in *Proc. Theory and Practice of Computer Graphics*, 2003, pp. 726–733.
[19] M. Flagg, A. Nakazaway, Q. Zhangz, S. B. Kang, Y. K. Ryu, I. Essa, and J. M. Rehg, "Human video textures," in *Proc. 2009 Symp. Interactive 3D Graphics and Games*, 2009, pp. 199–206.
[20] Y. Kitamura, R. Rong, Y. Hirano, K. Asai, and F. Kishino, "Video agent: Interactive autonomous agents generated from real-world creatures," in *Proc. 2008 ACM Symp. Virtual Reality Software and Technology*, 2008, pp. 30–38.
[21] D. Huttenlocher, G. Klanderman, and W. Rucklidge, "Comparing images using the Hausdorff-distance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, pp. 850–863, 1993.
[22] S. M. Seitz and C. R. Dyer, "View morphing," in *Proc. 23rd Annu. Conf. Computer Graphics and Interactive Techniques*, 1996, pp. 21–30.
[23] P. Gao and T. W. Sederberg, "A work minimization approach to image morphing," *Vis. Comput.*, pp. 390–400, 1998.
[24] K. Mikolajczyk and C. Schmid, "Scale and affine invariant interest point detectors," *Int. J. Comput. Vis.*, pp. 63–86, 2004.
[25] D. Lowe, "Distinctive image features from scale-invariant keypoint," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
[26] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," in *Proc. ACM SIGGRAPH 03*, 2003, pp. 313–318.
[27] Y. Horry, K.-I. Anjyo, and K. Arai, "Tour into the picture: Using a spidery mesh interface to make animation from a single image," in *Proc. 24th Annu. Conf. Computer Graphics and Interactive Techniques*, 1997, pp. 225–232.
[28] H. W. Kang, S. H. Pyo, K.-I. Anjyo, and S. Y. Shin, "Tour into the picture using a vanishing line and its extension to panoramic images," *Comput. Graph. Forum*, vol. 20, no. 3, pp. 132–141, 2001.
[29] J. Han, D. Farin, and P. H. N. de With, "Broadcast court-net sports video analysis using fast 3-D camera modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 11, pp. 1628–1638, 2008.
[30] C. Kenner, Glovepie, 2007. [Online]. Available: http://carl.kenner.googlepages.com/glovepie_download.
[31] WiiLi, WiinDows, and WiiBrew, "Wiiyourself", 2009, 2009 [Online]. Available: http://wiiyourself.gl.tter.org/
[32] T. Schlomer, B. Poppinga, N. Henze, and S. Boll, "Gesture recognition with a wii controller," in *Proc. 2nd Int. Conf. Tangible and Embedded Interaction*, 2008, pp. 11–14.
[33] C.-C. Chang and C.-J. Lin, Libsvm: A Library for Support Vector Machines, 2009. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm.
[34] Y.-T. Wang, Y.-N. Liu, S.-Y. Chien, Y.-H. Yang, Y.-L. Chen, and S.-L. Yeh, "A psychophysical analysis on perceptual limitation of motion image artifact reduction using 120 hz displays," in *Proc. SID Int. Symp. (SID 2009)*, 2009, pp. 1223–1226.
[35] J.-H. Lai, P.-C. Wu, C.-L. Chen, C.-C. Kao, and S.-Y. Chien, "Tennis real play," in *Proc. Int. Conf. Consumer Electronics*, 2011, pp. 275–276.
[36] T. Igarashi, T. Moscovich, and J. F. Hughes, "As-rigid-as-possible shape manipulation," in *Proc. ACM SIGGRAPH 05*, 2005, pp. 1134–1141.
[37] D. Mahajan, F.-C. Huang, W. Matusik, R. Ramamoorthi, and P. Belhumeur, "Moving gradients: A path-based method for plausible image interpolation," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 42:1–42:11, 2009.
[38] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 689–694, 2004.
[39] J.-H. Lai, C.-L. Chen, and S.-Y. Chien, "Architecture design and analysis of image-based rendering engine," in *Proc. Int. Conf. Multimedia and Expo.*, 2011, pp. 1–6.

Jui-Hsin Lai received the B.S. degree in electronics engineering from the National Chiao-Tung University (NCTU), Hsinchu, Taiwan, in 2005. In 2011, he received the Ph.D. degree from the Graduate Institute of Electronics Engineering at National Taiwan University (NTU), Taipei, Taiwan.

During 2007 to 2011, he was a project leader in Yotta-Labs, an IC design house in Taipei, for designing the algorithm and hardware architecture of vision-based object tracking, face detection, and recognition. Since 2011, he was been a post-doctoral fellow in the Graduate Institute of Networking and Multimedia, NTU. His research interests include the applications of interactive multimedia, computer vision, sports video, video/image processing, and VLSI architecture design of multimedia processing.

Chieh-Li Chen received the B.S. degree in electrical engineering from National Taiwan University (NTU), Taipei, Taiwan, in 2008, and the M.S. degree in electronics engineering from the Graduate Institute of Electronics Engineering at NTU in 2010. She is pursuing the Ph.D. degree in the Department of Bioengineering, University of Pittsburgh, Pittsburgh, PA.

She works as a graduate student researcher in the Ophthalmic Imaging Research Laboratory [Glaucoma Imaging Group (GIG)]. With the interests in image processing, she is currently working on developing image processing software to improve the understanding and help the analysis of glaucoma imaging obtained from optical coherence tomography (OCT) devices.

Po-Chen Wu received the B.S. degree from the Department of Electrical Engineering, National Taiwan University (NTU), Taipei, Taiwan, in 2010. Since 2011, he has been pursuing the Ph.D. degree in the Media IC and System Laboratory, Graduate Institute of Electronics Engineering, National Taiwan University.

His research interests include the applications of computer vision, image processing, and augmented reality.

Chieh-Chi Kao received the B.S. degree in electrical engineering, from the National Taiwan University (NTU), Taipei, Taiwan, in 2009. He is currently pursuing the M.S. degree in the Media IC and System Laboratory, Graduate Institute of Electronics Engineering, NTU.

His research interests are in computer vision, machine learning, multimedia analysis, and related VLSI architecture.

Min-Chun Hu (M'11) received the B.S. and M.S. degrees in computer science and information engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2004 and 2006, respectively. In 2011, she received the Ph.D. degree in the Graduate Institute of Networking and Multimedia, National Taiwan University, Taipei, Taiwan.

Currently, she is a postdoctoral research fellow of the Research Center for Information Technology Innovation, Academia Sinica. Her research interests include digital signal processing, digital content analysis, pattern recognition, computer vision, and multimedia information retrieval.

Shao-Yi Chien (S'99–M'04) received the B.S. and Ph.D. degrees from the Department of Electrical Engineering, National Taiwan University (NTU), Taipei, Taiwan, in 1999 and 2003, respectively.

During 2003 to 2004, he was a research staff in Quanta Research Institute, Tao Yuan County, Taiwan. In 2004, he joined the Graduate Institute of Electronics Engineering and Department of Electrical Engineering, National Taiwan University, as an Assistant Professor. Since 2008, he has been an Associate Professor. His research interests include video segmentation algorithm, intelligent video coding technology, perceptual coding technology, image processing for digital still cameras and display devices, computer graphics, and the associated VLSI and processor architectures. He has published more than 170 papers in these areas.

Dr. Chien serves as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY and *Springer Circuits, Systems and Signal Processing* (CSSP). He also served as a Guest Editor for *Springer Journal of Signal Processing Systems* in 2008. He also serves on the technical program committees of several conferences, such as ISCAS, ICME, A-SSCC, SiPS, and VLSI-DAT.