ABSTRACT

Ball-detection-and-tracking in broadcast tennis video (BTV) is a crucial but challenging task in tennis video semantics analysis. Informally, the challenges are due to camera motion and the other causes such as the presence of many ball-like objects and the small size of the tennis ball. The trajectory-based approach proposed by us in our previous papers mainly counteracted the challenges imposed by causes other than camera motion and achieves a good performance. This paper proposes an improved trajectory-based ball detection and tracking algorithm in BTV with the aid of homography, which counteracts the challenges caused by camera motion and bring us multiple new merits. Firstly, it acquires an accurate homography, which transforms each frame into the “standard” frame. Secondly, it achieved higher accuracy of ball identification. Thirdly, it obtains the ball projection position in the real world, instead of ball location in the image. Lastly, it also identifies landing frames and positions of the ball. The experimental results show that the improved algorithm can obtain not only higher accuracy in ball identification and in ball position alike, but also ball landing frames and positions. With the intent of using homography to improve the video-based event detection for smart home we also do some experiments on acquiring the homography for home surveillance video.

Keywords: Tennis video, Ball detection and tracking, Trajectory-based, Hitting, Ball landings, Homography.

1. INTRODUCTION

Sports video semantics analysis has been a hot topic due to its wide spectrum of applications and huge potential commercial value. Since the “ball” is the attention focus of viewers and players in tennis, ball-detection-and-tracking becomes a crucial task in tennis video analysis. However, it is a very challenging problem due to camera motion and other causes such as the presence of many ball-like objects, and the small size and the high speed of the ball. Furthermore, these causes exacerbate each other. Some of these challenges are shared by ball-game videos; some are peculiar to broadcast tennis video. Our previous paper developed a trajectory-based ball detection and tracking algorithm for broadcast tennis video (BTV), which mainly alleviates the challenges due to causes such as the presence of many ball-like objects, the small ball size, etc., and achieved a good performance. However, the algorithm can only obtain the ball locations in image space rather than the ball projection positions in the real world because it does not do any camera-related computation.

This paper proposes an improved trajectory-based ball detection and tracking algorithm in broadcast tennis video (BTV), which can obtain the ball projection positions and achieve an even higher accuracy in identifying the ball. In addition, it can detect ball landing position. These improvements mainly benefit from partially recovering the effect of camera motion by homography, which transforms each frame into the standard frame, whose lookat is the cluster center of all lookats of all the frames in the considered clip. The lookat of frame is a point in the real world that corresponds to the center of the frame. Essentially, homography brings us two benefits. The first is that it transforms the image position of the ball into its projection position. The other is that homography makes the ball trajectory (ball motion
trace) to be the trajectory that is consistent with the physical law, resulting in a trajectory that can be depicted by an analytical function. The approach used in this paper, which improves the video analysis by making use of homography, can be applied to many other applications such as video surveillance, soccer ball detection and tracking, and video segmentation. For example, in video surveillance homography can help us to get more accurate human moving trace which gives more accurate results for human behavior analysis.

The proposed algorithm in this paper comprises five components as depicted in Fig 1. The first component is to find an accurate homograph matrix to transform each frame to the standard frame. This step first computes the homography based on the features detected. Then it uses a Hough-like search procedure to acquire a more accurate homography. The other four components have appeared in our previous paper\(^{12}\). However, the ways of implementing them have been improved. For example, in the trajectory processing component we not only find more accurate ball positions, but also detect ball landing frames and find ball landing positions.

The algorithm presented in this paper works on clips, where a clip is a sequence of consecutive frames showing the tennis court in the given video. Based on the fact that tennis court scenes are mainly interleaved by non-court scenes such as zoom-in shots of the players or the audience, a preprocessing procedure segments video into clips based on the dominant color arising from the presence of the tennis court in the tennis video. The proposed algorithm is to find not only the projection locations of the ball on the ground for each frame but also the landing positions of the ball in a clip. In this algorithm, we use homography to reduce the effect of camera motion, so that we can achieve better performance.

Related work lies in acquisition and application of homography as well as ball detection and tracking. In acquisition and application of homography, Dirk et al proposed a generic homography acquisition algorithm that can be applied to multiple types of sports videos in which the playfields consist of straight lines on plane by configuration of the playfield model\(^{12}\). For the first frame of a sequence, a robust court fitting procedure considers all combinations of straight lines; for the subsequent frames, a court-model based tracking algorithm is applied. However, the homography is acquired based on feature points. Hence, there is room to reduce the reprojection disparity between the court computed based on homography acquired and the image court in original frame. Iwase and Saito used homography between two camera views to estimate the player location if a player is occluded in the view in which he is being tracked\(^{5}\). In this way, they can achieve better performance of player tracking because they counteract most cases of occlusions. In ball detection and tracking, Yu et al first developed a trajectory-based algorithm for ball detection and tracking in broadcast soccer video and achieved a much better performance than the direct-object-detection ones\(^{13-14}\). Then Yu et al developed a trajectory-based algorithm for ball detection and tracking in broadcast tennis video and also achieved a good performance\(^{12}\). This trajectory-based approach has mainly counteracted the challenges caused by the high speed of the ball, ball merging, the presence of many ball-like objects, etc. These challenges, their causes, and their effects are discussed in our previous papers\(^{12-14}\). However, the trajectory-based approach\(^{12-14}\) did not resolve the effect of camera motion very much. They can be improved by homography which makes the ball candidates form better trajectories. In addition, our previous algorithm for tennis ball detection and tracking\(^{12}\) can obtain only ball locations in the image, instead of ball projection positions on the ground.

We continue this paper with presenting our improved ball detection and tracking algorithm in Section 2. Section 3 gives the various experimental results. We conclude our paper in Section 4.

---

**Fig 1.** Block diagram of the improved tennis-ball detection and tracking algorithm for broadcast tennis video.
2. BALL DETECTION AND TRACKING

This section explains the components of the proposed algorithm depicted in Fig 1 in turn. The components which appeared in our previous paper\textsuperscript{12} will be described concisely, whereas the role of the homograph transform in the proposed algorithm will be discussed in greater detail.

2.1. Acquiring Accurate Homography

Here we present a procedure to acquire an accurate homograph matrix for transforming each frame into the “standard” frame. Essentially, we use the entire segmented court from a frame not just the feature points to compute an accurate homograph matrix for the frame. As depicted in Fig 2, this procedure takes two steps. In Step 1, it first segments the court from the given frame, detects the image court that consists of nine straight lines, and then extracts features (or feature points) that appearing in the frame among all 14 features on the ground for each frame, as depicted in Fig 3. In Step 2, it first computes an initial homograph matrix. Then it tunes the matrix computed against the defined measure function to reduce the reprojection disparity. Lastly it transforms the frame into the “standard” frame using the tuned homography. The “standard” frame is the frame whose lookat is close to the cluster center of lookats of all frames in the considered clip.

2.1.1. Feature Point Extraction

Court Segmentation: The court segmentation procedure separates those pixels belonging to the tennis court lines from other pixels. We first find the court color range by statistical analysis. Then we paint all the pixels in this range with a single color, termed as the court color. Next we find the lines separating the audience from the playing field by detecting the change pattern of color for each row and column of the image. We then paint the audience area in the court color. Thus, most of the pixels not in the court color are the pixels on the segmented court, comprising nine straight lines, the net, and players. An example of a segmented tennis court is shown in Fig 4. We do not remove the net and the players because they will not affect our subsequent computation.

Straight Line Detection: The segmented court on the ground consists of nine straight lines. We use the gridding Hough transform (GHT) which is developed in our previous work\textsuperscript{11} to detect these lines. The basic idea is to grid the

Fig 4. A sample frame showing the segmented court.

Fig 5. The best homograph matching and transformed standard frame for a sample frame.
frame using gridding lines, which will cross the long straight lines in multiple points. The pairs of these crossing points within each grid will tell us the parameters of the straight lines in the frame.

Court Fitting: Once we have obtained the straight lines, we fit these lines to the ground model of the tennis court shown in Fig 3. In our fitting procedure, we detect the net and use the net as a reference to guide our matching of the straight lines in the image and the straight lines in the ground model. Within a small angular range around the horizontal direction, we first look for two horizontal straight lines below the net and two lines above the net and then match them to the corresponding physical lines. Similarly, in the vertical direction, we look for the middle line and then the left and the right lines. Then we match them to the corresponding physical lines. Note that though most of frames will get the correct fitting, some of them may go wrong due to causes such as missing lines. To resolve this issue, we identify the frames with wrong fitting by clip-wise analysis of the lookats of all frames (the lookat of frame is a point in the real world that corresponds to the center of the frame). The lookats of frames will form a relatively smooth function. When a frame is fitted wrongly, the lookat computed for this frame will not be around this function. For these frames, we use the feature points of their neighbors. Fortunately, there are very few such frames.

Ground Features: Once we have fitted the ground model of the tennis court we can find the image coordinates of the points C1 to C14 shown in Fig 3 by finding the intersection points of the straight lines detected. These form our initial ground features. Inevitable deviation of the features detected will be reflected in a big value of the reprojection disparity. The following Hough-like tuning procedure will significantly reduce the reprojection disparity.

2.1.2. Homography

Initial Matrix: Once feature points are found, we can determine the homograph matrix given in the following formula (1) (see pp 87-91 in [2]), which transforms an image point (represented by a homogenous 3-vector) \( X = (x_1, x_2, 1) \) to a point \( X' = (x'_1, x'_2, 1) \) in another image.

\[
X' = HX. 
\] (1)

Disparity Measure of Two Court Images: Let \( C_{\text{std}} \) and \( C_{\text{com}} \) be any two images of the tennis court (ground portion). We define a line-based function to measure the disparity of \( C_{\text{std}} \) and \( C_{\text{com}} \). We use L1 to L9 to denote the straight lines as shown in Fig 3. We use \( L_{i, \text{std}} \) and \( L_{i, \text{com}} \) to denote the sets of all the points contained in the straight line \( L_i \) in \( C_{\text{std}} \) and \( C_{\text{img}} \), respectively for \( i = 1 \) to 9. The measure function \( M_L(i) \) of straight line \( L_i \) is defined as

\[
M_L(i) = \frac{|L_{i, \text{std}} \cap L_{i, \text{com}}|}{|L_{i, \text{std}}|} \times 100. 
\] (2)

where \( |\cdot| \) is the cardinality of a set.

The measure function \( M_{\text{cod}}(C_{\text{std}}, C_{\text{com}}) \) for measuring the disparity of the two courts is defined as

\[
M_{\text{cod}}(C_{\text{std}}, C_{\text{com}}) = \frac{1}{9} \sum_{i=1}^{9} \psi(i). 
\] (3)

where \( \psi(i) = \begin{cases} M_L(i) & \text{if } M_L(i) \geq \beta; \\ 0 & \text{otherwise.} \end{cases} \) and \( \beta \) is a predefined threshold.

Measure Function: Let \( C_{\text{std}} \) be the court in the standard frame and \( C_{\text{img}} \) be the segmented court in frame F. We use \( C_{\text{inf}}(H, C_{\text{img}}) \) to denote the transformed court from \( C_{\text{img}} \) via \( H \) according to equation (1). For given \( H \) and \( F \), the measure function \( M(H, F) \) is defined as

\[
M(H, F) = M_{\text{cod}}(C_{\text{std}}, C_{\text{inf}}). 
\] (4)

Tuning of Homography: Let \( H_0 = \begin{pmatrix} h_{0}^{0} \\ h_{0}^{y} \\ h_{0}^{x} \end{pmatrix} \) be the homograph matrix computed based on feature points of the frame F. Then we prepare a small Hough space \( H_{sp} = \left\{ H = \begin{pmatrix} h_{ij}^{0} \\ h_{ij}^{y} \\ h_{ij}^{x} \end{pmatrix} : h_{ij}^{y} = h_{ij}^{0} \pm k \times g_{ij}, k = 1, 2, \ldots, N \right\} \) enclosing it, where \( g_{ij} \) is the
search interval. We compute $M(H, F)$ one by one for all $Hs$ in $Hsp$ and keep the best homograph matrix. This is captured in Procedure 1.

**Procedure 1**

**Step 0:** Input initial homograph matrix $H_0$ and segmented court.

**Step 1:** Form the Hough space $Hsp$ enclosing $H_0$.

**Step 2:** Initialize $V_{max} = M(H_0, F)$; $H_{td} = H_0$.

For all $H$ in $Hsp$ do
- If $M(H, F) > V_{max}$, then $V_{max} = M(H, F)$; $H_{td} = H$.

Terminate and output $H_{td}$.

**Frame Transform:** Once we obtain the best homograph matrix $H_f$ for a frame $F$ we transform $F$ into the standard frame $F_{std}$ according to $H_f$, as illustrated in Fig 5. An unknown pixel in the transformed frame, which corresponds to a pixel out of the view in the original frame, is given the court color. After the transformation, all frames in a clip are the frames with the same camera view.

### 2.2. Ball Locations in Hitting Frames

We first do two tasks: (a) finding positions of two players; and (b) finding hitting frames of the given clip using the method proposed by Xu *et al.* Then, we find the location of the ball in each hitting frame by finding the positions of the rackets. Note that the found location of the racket may not be accurate. We, however, only need these locations to guide us to find the ball trajectories, not the final ball location. The final ball locations are computed based on the ball trajectories acquired.

### 2.3. Ball Candidate Detection

Here we explain how to obtain the ball candidates in the standard frame that is transformed from the original frame via homography. It is difficult to identify the ball within the frame because some of the non-ball objects look like the ball in a frame. We use sieves to remove as many non-ball objects as possible in each frame. Then the remaining objects are considered as the ball candidates of the frame. We use this anti-model approach to obtain the ball candidates because it is hard to build the representation of the ball. The following are the sieves to remove the non-ball objects.

**Court Sieve** $\Theta_1$: We filter out audience area because the playing ball will not appear in that area. We also filter out the court lines. We can identify these two targets exactly as we know where they are in the transformed standard frame.

**Ball Size Sieve** $\Theta_2$: We filter out the objects out of the ball-size range, which are estimated based on homography. We can acquire a homography from the physical ground model to the standard frame because we know the coordinates of the features in the real world. Thus, we can calculate the ball size at any pixel in the standard frame via this homography since we know the size of the physical tennis ball. The real size of the ball in the image might be different from the estimated size due to various reasons such as the ball is not on the ground but higher up in the air. We use a range of allowable ball sizes to cope with this difference.

**Ball Color Sieve** $\Theta_3$: We filter out the objects with too few ball color pixels.

**Shape Sieve** $\Theta_4$: We filter out the objects out of the range of width-to-height ratio.

Each sieve $\Theta_i$ is a Boolean function on domain $O(F) = \{o : o$ is an object in the field of frame $F\}$.

$$\Theta_i(o) = \begin{cases} 0 & \text{if sieve } \Theta_i \text{ removes it,} \\ 1 & \text{otherwise.} \end{cases}$$

(5)

After sieving, the remaining objects of $C(F) = \{o : o \in O(F), \Theta_i(o) = 1 \text{ for } i = 1 \text{ to } 4\}$ form the ball candidate set $C(F)$ of frame $F$. 

---

Downloaded from SPIE Digital Library on 29 Sep 2010 to 157.193.213.84. Terms of Use: http://spiedl.org/terms
Candidate Classification: We use a rule-based approach to classify the candidates into three categories. In the procedure of candidate classification, we evaluate candidates by four features: size, color, and distance from other objects. A measurement is defined on a feature as follows.

\[
\begin{align*}
S(o) &= (R - r); \\
K(o) &= n : N; \\
D(o) &= \text{distance from the object to other objects.}
\end{align*}
\]

where \( r \) is the radius of the object, \( R \) is the estimated ball radius, \( n \) is the number of pixels in ball color range, \( N \) is the number of all the pixels in the object, and \( D(o) \) is the distance from the object to other objects. The distance between two objects is defined as the minimum of all the distances from a pixel in one object to a pixel in the other.

The ball-candidates that meet all three above criteria are classified into Category 1, while the others are placed in Category 2. The candidates in Category 1 are more likely to be the ball than those in Category 2. This classification result will be used in evaluating the trajectory in Section 2.5.

2.4. Candidate Trajectory Generation

It has been noted that within a frame \( F \), it is very difficult to identify the ball from among the ball-candidates in \( C(F) \). To resolve this difficulty, it is much better to study the trajectory information of the ball since the ball is the “most active” object in tennis video.

We have partitioned the entire video into the sequences \( N_0S_1S_2\ldots N_nS_n \), where each \( S_k \) is a sequence of contiguous frames that contain the tennis court, and each \( N_k \), is a sequence of contiguous frames that do not contain the tennis court.

The input of our algorithm is one of the court sequences (clips), say \( S_k \). For this \( S_k \) we aim to generate a set, \( \Gamma(S_k) \), of candidate-trajectories for each video sequence \( S_k \). We start with some definitions. Let \( F_1, F_2, \ldots, F_n \) denote the frames in a video sequence \( S_k \) and let \( C(F_i) \) be the set of ball-candidates for the frame \( F_i \). For each ball-candidate \( c \) in \( C(F_i) \), the location is denoted by \((x, y)\), with \((0, 0)\) being the bottom-left corner of a frame.

Y-Time Plots: To “visualize” the motion of the ball-candidates in the video sequence \( S_k \), we first plot some “feature” of the ball-candidates over time (as represented by the frame number \( i \)). We call these plots Candidate Feature Plots (CFPs). For example, if the “feature” selected is the \( y \)-coordinate, then we call the resulting plot the CFP based on \( y \) coordinate (CFP-\( y \)) for \( S_k \). In CFP-\( y \), each ball candidate \( c \) in \( C(F_i) \) is represented by the point \((i, y)\). Fig 6 is a CFP-\( y \) and more examples can be found in [7-8].

Location-Time Plots: For trajectory generation, we use the CFP based on location (CFP-I) in which we plot the location of the ball-candidates over time. In the CFP-I, each ball-candidate \( c \) in \( C(F_i) \) with location \((x, y)\) in the frame \( F_i \) is represented by the point \((i, x, y)\). The CFP-I is 3-dimensional and has more discriminating power when identifying
trajectories, but the \textit{CFP-y} is easier to visualize on paper. We use the \textit{CFP-y} to illustrate our ideas, but our algorithms use the \textit{CFP-l} for trajectory processing.

We also plot hittings and landings in our CFPs. In Fig 6, blue, grey, and thin grey vertical lines indicate frames where near-player hittings, far-player hittings, and landings occur respectively. The near-player and far-player are the players close to and far from the camera in frame, respectively.

\textbf{Candidate Trajectory Generation Algorithm:} To generate the list of candidate trajectories from the \textit{CFP-l}, we use a trajectory growing algorithm that extends a trajectory based on the motion of the projection objects in the \textit{CFP-l}. See the details of the algorithm in our previous papers\cite{12-14}.

\subsection*{2.5. Trajectory Processing}

\textbf{Trajectory Confidence Index:} Let \( \Gamma = \{ T : \text{candidate trajectory} \} \) be the trajectory set of a given video clip in the candidate feature image. Let \( \lambda_1, \lambda_2, \ldots, \lambda_m \) be all properties of trajectory \( T \). These properties characterize the statistical properties of objects in \( T \) (such as percentage of candidates in category 1, the length of the trajectory, etc). A function \( \Omega(\lambda_i) \) computes the confidence index of \( T \) being a ball trajectory with respect to \( \lambda_i \). The confidence index \( \Omega(T) \) of \( T \) being a ball trajectory is defined below:

\begin{equation}
\Omega(T) = \sum_{i=1}^{m} \Omega_i(\lambda_i).
\end{equation}

\textbf{Trajectory Discrimination:} Based on the confidence index and the fact that there is at most one ball in a frame, we formed a procedure to select ball trajectories\cite{12-14}.

\textbf{Ball Projection Location Computation:} When the ball flies near the far-player, it may be occluded by the player, or too small to be seen. Thus, it is useless to track the ball. Hence, we compute the ball location by extending the trajectory to the far-player’s hitting location. As shown in Fig 6, a ball trajectory is a smooth curve because we have transformed each frame into the standard frame. Here we use a cubic curve to \textit{locally} fit the ball trajectory. We use the identified balls to determine the coefficients of the following expression:

\begin{equation}
y = an^3 + bn^2 + cn + d.
\end{equation}

After acquiring the above equation, we use it to estimate the projection positions of the missing ball.

\textbf{Ball Landing Detection:} Once we obtain the ball trajectories as shown in Fig 6 we form a ball position function against frame number \( i \) as below for a clip.

\begin{equation}
y = f(i).
\end{equation}

Then we compute \( f'(i) \) (the derivative of \( f(i) \)) and find the frame with the maximum of \( f'(i) \) between each pair of hittings, i.e. landing frame. The ball position in a landing frame is its landing position. Of course, the smooth \( f'(i) \) indicates there is no landing.

\textbf{Discussion:} Fig 7 gives the ball trajectories of the same clip as that given in Fig 6, but in Fig 7, the ball locations are not transformed. Generally, the trajectories in Fig 7 are similar to those in Fig 6. However, some real balls in Fig 7 are not on the trajectory. This explains why the algorithm in \cite{12} cannot achieve 100\% accuracy in locating the ball.

\section*{3. EXPERIMENTAL RESULTS}

The proposed algorithm has been tested on 5 clips, which are extracted from mpeg2 704x576 video recorded from TV signal. In terms of running time, the proposed algorithm \textit{ALG}_{new} is much longer than the previous algorithm \textit{ALG}_{old}-
For acquiring ball candidates, the average time of ALG\textsubscript{new} for a frame is 86.15s on a P4/1.7Ghz PC with 512MB RAM; that of ALG\textsubscript{old} is 19.21s.

Fig 8 gives a sample frame showing that our Hough-like tuning can visually reduce the disparity between the generated court and the image court in the original frame. In Table 1, \(V_{\text{init}}\) and \(V_{\text{tune}}\) rows are the average values of the measure function, defined in equation (4), of all initial and tuned homograph matrices of a clip, respectively. \(P_{\text{better}}\) is the average percentage by which \(V_{\text{tune}}\) has improved from \(V_{\text{init}}\) for the clips. Table 1 shows that our Hough-like tuning procedure has numerically improved the homograph matrices.

### Table 1. Comparison of measure function values of initial and tuned homograph matrices

<table>
<thead>
<tr>
<th>CLIP</th>
<th>clip1</th>
<th>clip2</th>
<th>clip3</th>
<th>clip4</th>
<th>clip5</th>
</tr>
</thead>
<tbody>
<tr>
<td>(V_{\text{init}})</td>
<td>0.5516</td>
<td>0.3922</td>
<td>0.5808</td>
<td>0.5970</td>
<td>0.6684</td>
</tr>
<tr>
<td>(V_{\text{tuned}})</td>
<td>0.7665</td>
<td>0.5468</td>
<td>0.7768</td>
<td>0.7754</td>
<td>0.8105</td>
</tr>
<tr>
<td>(P_{\text{better}})</td>
<td>38.96%</td>
<td>39.42%</td>
<td>33.75%</td>
<td>29.88%</td>
<td>21.26%</td>
</tr>
</tbody>
</table>

### Table 2. Comparison of accuracy of ball location (in pixel)

<table>
<thead>
<tr>
<th>CLIP</th>
<th>clip1</th>
<th>clip2</th>
<th>clip3</th>
<th>clip4</th>
<th>clip5</th>
</tr>
</thead>
<tbody>
<tr>
<td>(D_{\text{new}})</td>
<td>1.53</td>
<td>1.82</td>
<td>1.74</td>
<td>1.87</td>
<td>1.47</td>
</tr>
<tr>
<td>(D_{\text{avg}})</td>
<td>0.52</td>
<td>0.49</td>
<td>0.61</td>
<td>0.47</td>
<td>0.53</td>
</tr>
</tbody>
</table>

Fig 8. Feature points acquired by computing the crossings of straight lines and computed based on the tuned homography as well as the generated courts based on each set of feature points.
Our previous algorithm\textsuperscript{12} still misses some balls that are not on any trajectory as illustrated in Fig 7. After we transform each frame into the standard frame, all balls are on trajectory. Hence, ALG\textsubscript{new} in this paper achieves 100\% accuracy in identifying the ball locations. Another merit of the improved algorithm is that it can obtain more accurate ball locations. In Table 2, $D_{prAlg}$ and $D_{new}$ rows give the average discrepancy of all detected balls from the groundtruth for ALG\textsubscript{old} in \cite{12} and ALG\textsubscript{new}. Table 2 shows that the algorithm in this paper has improved the ball location accuracy of the detected balls.

The accurate homography also helps us acquire accurate intrinsic and extrinsic camera parameters which have many applications. For example, based on these accurate camera parameters, we can insert 3D projected virtual content into broadcast tennis video, this is better than image-based insertion\textsuperscript{15}. Two sample frames with inserted 3D projected virtual content are given in Fig 9 and Fig 10.

With the encouragement of the success that the homography has improved the performance of the ball detection and tracking algorithm for broadcast tennis video, we are using homography to improve video-based event detection for the smart home. Here we give a sample frame to show how to acquire homography of home surveillance video. In home surveillance video, the architecture of the house and the geometry of articles can be assumed to be known. From the known architecture, we can find the feature points for acquiring homography. For example, in Fig 11, we are able to acquire the homography because we know the geometry of the carpet.

4. CONCLUSION AND FUTURE WORK

We have presented a trajectory-based ball detection and tracking algorithm for broadcast tennis video, which is an improved version of our previous algorithm\textsuperscript{12}. The previous algorithm mainly alleviated the challenges raised by causes besides camera motion such as the presence of many ball-like objects and the small size of the tennis ball, whereas the
algorithm presented in this paper additionally counteracts the challenges brought to us by the camera motion. It first finds accurate homography to transform each frame into the standard frame of the considered clip. It is shown that the homography component greatly improves the performance of trajectory-based ball detection and tracking algorithm in the multiple aspects. It not only increases the accuracy in identifying the ball, but also enhances the accuracy in determining the ball projection position. In addition, it detects ball landing frames and landing positions based on the accurate ball trajectory. The contributions of this paper are two-fold: it develops a procedure to robustly acquire an accurate homograph matrix of each frame; and it forms an improved version of ball detection and tracking algorithm, which achieves better performance than our previous algorithm\textsuperscript{12}. Two of many remaining jobs in tennis video analysis are as follows. First, we will further evolve the algorithm presented in this paper into an end-to-end system for ball detection and tracking of broadcast tennis video. Second, we will analyze the tactics of players and winning-patterns\textsuperscript{9}, and hence produce rich indexing of broadcast tennis video by making use of the ball position. In acquisition and application of homography we will use homography in video-based event detection for smart homes to enhance the accuracy of event detection.

5. REFERENCES