Tracking the soccer ball using multiple fixed cameras

Jinchang Ren, James Orwell, Graeme A. Jones, Ming Xu

1. Introduction

The convergence of computer vision and multimedia technologies, in particular high-speed cameras and networking, has led to opportunities to develop applications for automatic sports analysis, especially soccer video analysis, including content-based indexing, retrieval and visualization [1–3]. Other relevant applications are those analyses of golf [4], tennis [5], American football [6], hockey [7], baseball [8] and basketball [9] as well as ping pong and cricket [10], etc. Through image and motion analysis, additional information can be extracted for better comprehension of video and sports contents, such as video content annotation, summarization, team strategy analysis, and verification of referee decisions, as well as further 2D/3D reconstruction and visualization [11–16].

In any ball game like soccer match, the ball is invariably the focus of attention. Although players can be successfully detected and tracked on the basis of color and shape [1,3,13,16], similar methods cannot be extended to ball detection and tracking for several reasons:

- The ball is relatively small and moves fast, and consequently exhibits variable size, and motion-blurred shape and color when moving at speed (see examples in Fig. 1);
- It is a difficult task to track the ball when it is occluded or ‘possessed’ by players;
- There are many false alarms similar to the ball, such as small regions near the field lines and regions of players’ bodies;
- The ball color is designed to be easily visible by players and spectators. It can have single or multiple colors. It may become discolored from contact with the playing surface, especially in rainy conditions. In the work presented in this paper, the assumption is that the ball surface is white (English Premiership matches). This assumption can be modified as necessary.

Although TV broadcast streams are the most common sources of soccer videos, there is sometimes also video sequences from fixed cameras available. In TV streams, the ball is mostly of good resolution in the image center. However, due to complex camera movements and partial views of the field, it is hard to obtain accurate camera parameters for off-field ball positioning. In Gong et al. [1], white color and circular shape are employed to detect balls in image sequences. In Yu et al. [15], candidate balls are first identified by size range, color and shape, and further verified using motion information obtained from a Kalman filter. In Yow et al. [2], ball detection is undertaken by template matching in each of the reference frames and then the ball is tracked between these frames. In Seo et al. [13], template matching and Kalman filtering are used to track balls after manual initialization. In Liang et al.
[17], color, size and shape features are also employed for ball detection, followed by graph-based filtering. In Tong et al. [18], an indirect strategy is employed for ball detection by eliminating non-ball objects using color and size features, however, it fails in dealing with cases of occlusion or small size of the ball. Since color and shape varies considerably in soccer games (see Fig. 1), this would suggest that further work is necessary to achieve a robust solution for this problem.

Using multiple fixed cameras has the advantage that it brings wider field of views to easily establish on-field positions for visualization. Bebie and Bieri [11] and Matsumoto et al. [12] used two and four cameras in their systems for soccer game reconstruction and optimized viewpoint determination, respectively. Ohno et al. [16] adopted eight cameras arranged on both sides of the field to attain a full view of the game. Although motion-based tracking models are introduced in [11] and [16], there is no given process to automatically identify the ball before tracking. In Matsumoto et al. [12] and D’Orazio et al. [14], template matching and a modified Hough transform are presented to detect balls in soccer videos, respectively. Since irregular ball shapes are usually extracted in different velocities, these two methods are still insufficient. In Choi and Seo [19], the ball is detected and tracked by removing players’ blobs in a so-called accumulated measurement. However, it cannot deal with occluded case and may fail when there are false alarms of similar color to the ball.

For 3D ball positioning, several model-based approaches have been presented. Bebie and H. Bieri [11], model 3D trajectory segments by Hermite spline curves. However, about one-fifth of the ball positions need to be set manually before estimation. In Matsumoto et al. [12], epipolar line constraints between multiple cameras are utilized. In Ohno et al. [16], the 3D ball trajectory is modeled by considering air friction and gravity but depends on an unsolved initial 3D velocity. In Kim et al. [20] and Reid and North [21], reference players and shadows are utilized in the estimation of 3D ball positions. However, this requires a dependable light source of known position, which is not always available.

In this paper, a comprehensive model-based methodology is proposed for ball detection and tracking from real soccer sequences, in which domain knowledge of soccer games is modeled as the base for further processing. The main highlights of our method can be summarized below. Firstly, ball filtering is performed on the output of Kalman trackers which allows velocity information to be employed in the classification stage. Also, a tracking plus matching process is utilized in solving occlusions when the ball is merged with players. Secondly, the expected appearance of a moving ball is explicitly modeled to improve the ball classification process. Thirdly, occlusion-reasoning and tracking-back is employed to recover any ball merged with players as well as to remove false alarms. Finally, a 3D trajectory model is introduced, and the estimated 3D ball positions are fed back into the earlier stages for more efficient and robust ball detection and tracking.

The remaining part of this paper is organized as follows. In Section 2, domain knowledge of soccer games is modeled. Then, Section 3 discusses foreground detection and tracking, in which a modified tracking plus matching process is introduced. Details on ball filtering with post-processing is presented in Section 4, and in Section 5, a 3D ball trajectory model is described, including approaches for 3D ball positioning and feedback in 2D ball detection and tracking. Experimental results and discussions with quantitative evaluations are given in Section 6, followed by a brief conclusion in Section 7.

2. Domain knowledge modeling of soccer games

Domain knowledge including color, ball shape and pitch geometry is widely adopted in soccer and other sports video analysis systems [1–3, 12, 14–16, 22]. In the TV broadcasting domain, knowledge about closed captions, audio, slow-motion replays and special zoom enables more specific models explored in shot detection and semantic indexing of sports videos [3, 22–25]. Like many other ball games, soccer contains rich semantic and contextual information. Consequently, domain knowledge of soccer can be used in modeling such scenarios for context-based vision [26], or closed world tracking [6]. According to Strat and Fischler [26] and Intillie and Bobick [6], context here refers to selection of knowledge for dynamic and multi-object tracking, and a closed world means a spatial–temporal region of known specific context. In closed world tracking, context-specific features have been proved robust and effective in a complex scene. In this paper, this approach has been applied to the soccer domain and the relevant context knowledge is modeled in several aspects and discussed as follows.

2.1. Field modeling

In real soccer games, the play field (or ‘pitch’) can be exactly modeled with many corner points of given locations. This pitch model is useful in camera calibration to obtain bi-directional coordinate transforms between an image-plane to the ground plane. In our system, Tsai’s algorithm for camera calibration [27] is adopted. In general, tracking in soccer domain is a wide-area surveillance problem; hence multiple cameras are normally required. In a multi-camera system, the 3D position and field-of-view (FOV) of each camera can be determined before tracking, together with boundaries and transition areas between different views. Consequently, white field lines within each camera view can be modeled as a mask image to remove potential false alarms caused by these lines. The FOVs of all eight cameras are shown in Fig. 2a. Fig. 2b and c illustrate an empty pitch and the field line mask from the view of camera #1. These field lines are detected using the Canny edge detector [28] and further linked with a morphological ‘closing’ operation.
2.2. Object modeling

The objects in soccer context include a single ball (in play at any time) and 25 persons comprising the referees and players. A model of the ball needs to successfully discriminate it from observations of other moving objects in the scene: the above persons, other people (e.g. warming up on the touchline), other objects (such as windblown litter and displaced turf) and also false observations such as pitch markings judged as motion from camera-shake.

Now further details about these objects are provided. The ball diameter \(d_0\) is between 0.216 m and 0.226 m, implying an area (projected into any direction of view in world co-ordinates) of about 0.04 m\(^2\). Its color depends on its condition and requirements from competitions; for the matches recorded for these experiments (English Premiership) its initial color is white. Normal minimum bounds on the player dimensions are: a height more than 1.7 m; a width of more than 0.25 m (with a corresponding minimum projected area of 0.45 m\(^2\)). On the basis of these dimensions, the ball can be clearly distinguished from the players and referees even though they sometimes appear in white, too. This clarity is not always achieved when using observations from the static cameras, as a consequence of the following factors:

1. The transformation from image dimensions to real (world) measurements requires an estimate of the distance of the object from the camera. This estimate can be inaccurate or missing.
2. The rapidly moving ball is subject to motion blur that increases its subtended image area and distorts its otherwise circular appearance.
3. Objects corresponding to regions of players or their uniforms (such as white socks) may be observed to have similar dimensions to the ball if they are separated from the other regions of the player (due to occlusion, image boundary condition, or imperfect segmentation)
4. The ball is frequently occluded from view by players.

Fig. 3. Shows example observations of the ball, players and player parts, field line false alarms that have been flagged by the motion detection process. Details on how to use the contextual knowledge above and classify these objects are discussed in Sections 3 and 4.

2.3. Motion modeling

As the focus of the game, the ball usually moves faster than players, which indicates the importance of motion features for ball detection and tracking. Indeed, the motion of the ball undergoes various phases when it is in play. For the ball that is in-play, an important distinction is between periods in which a player has or is attempting to have the ball under control; and periods in which it is moving freely between players. These states can be termed possessed and moving-free, respectively. When the ball is possessed it is frequently occluded by players. The moving-free ball can be sub-classified into flying and rolling phases, which in turn require tracking in 3D and 2D space. The ball that is out-of-play is less relevant to tracking. However, an important contextual cue is the use of replacement balls to accelerate the process of re-introducing the ball into play. This is an allowable discontinuity in the trajectory of ‘the ball’ (although outside the scope of this paper).

3. Detection and tracking of moving objects

In this Section, the method is described for tracking the moving objects in each of the multiple image planes. Image differencing is used to detect the objects, followed by Kalman filter-based tracking. For robustness, a two-stage adaptive background model is applied. In the first stage, a per-pixel Gaussian Mixture Model [29], \(\{\mu_k^{(l)}, \sigma_k^{(l)}, \omega_k^{(l)}\}\), is used to estimate an initial background, where \(\mu_k^{(l)}\), \(\sigma_k^{(l)}\) and \(\omega_k^{(l)}\) are the mean, root of the trace of covariance matrix, and weight of the \(l\)-th Gaussian distribution at frame \(k\). In the sec-
ond stage, this initial background image is continuously updated using a faster running average algorithm for efficiency [30]:
\[ \mathbf{F}_k = [a_k \mathbf{I}_k + (1 - a_k) \mathbf{F}_{k-1}] + [a_k \mathbf{I}_k + (1 - a_k) \mathbf{F}_{k-1}] \]
where 0 < a_k < 1, and \( \mathbf{F}_k \) refers to the foreground binary mask.

This method helps to slowly update the background image even in foreground regions.

Given the input image \( \mathbf{I}_k \), the foreground binary mask \( F_k \) can be decided by comparing \( | \mathbf{I}_k - \mathbf{F}_{k-1} | \) against a threshold. From the foreground masks, we can obtain a series of foreground regions representing candidate objects after a connected component analysis and thresholding by size. Each foreground region is represented by its centroid, bounding box and area. For each detected object, measurements from both image-plane and ground plane are obtained in pixels and meters, respectively. The former includes the bounding box of the foreground region and its centroid, and the latter contains its width, height and area determined from the foot of the foreground region. Besides, all detected small objects (less than 0.1 m²) are abandoned provided that their bounding boxes are overlapped with the field line mask.

3.1. Tracking in image-plane

An image-plane Kalman tracker is used to filter noisy measurements and split merged objects, in which the state \( \mathbf{x}_i \) and measurement \( \mathbf{z}_i \) are given by:
\[ \mathbf{x}_i(k + 1) = \mathbf{A}_i \mathbf{x}_i(k) + \mathbf{w}_i(k) \]
\[ \mathbf{z}_i(k) = \mathbf{H}_i \mathbf{x}_i(k) + \mathbf{v}_i(k) \]
where \( \mathbf{w}_i \) and \( \mathbf{v}_i \) are the image-plane process noise and measurement noise; \( \mathbf{A}_i \) and \( \mathbf{H}_i \) are the corresponding state transition matrix and measurement matrix defined in (4), with \( \Delta T \) denoting the time interval between two successive frames (for image formation); \( \mathbf{I}_2 \) and \( \mathbf{O}_2 \) represent \( 2 \times 2 \) identity and zero metrics.

The Mahalanobis distance is used to associate each observation to (at most) one tracked object, and the distance between the \( i \)th observation \( \mathbf{O}_i \) and the \( j \)th predicted object \( \mathbf{O}_j \) is given in Eq. 6, where \( \Sigma \) is the covariance matrix.
\[ \delta^2_{ij} = (\mathbf{O}_i - \mathbf{O}_j)^T \Sigma^{-1} (\mathbf{O}_i - \mathbf{O}_j) \]

3.2. Tracking correction

The tracking method described above is an effective and robust method for dealing with partial occlusions when the bounding box of two objects are overlapped in part, and further details on the procedure for splitting two merged objects can be found in [31]. However, if the bounding box of one (smaller) object, such as the ball, is completely contained by another (larger) object’s bounding box, then there is no valid observation for the ball, and the estimated state will be updated on prediction only. This will frequently lead to tracking failure. Since this kind of full occlusion happens quite often between the ball and players, it is necessary to solve this problem for robust tracking of the ball.

In the proposed scheme, an improved tracking plus matching process is introduced whenever a small object (of area less than 0.3 m²) is found being merged with another (larger) object of area more than 0.4 m². These two thresholds are defined according to the model in Section 2.2, allowing for an inaccurate measurement of ball size when it elevated from the ground plane. Then, a template of the small object is extracted before it is merged, which is used to find an optimal matching in the merged block.

Let \( r_{-1} \) and \( g_i \) be the template image and the merged block in frame \( k - 1 \) and \( k \), respectively, then their distance \( E(\Delta_i \Delta_j) \) is defined as follows, where the optimal matching of \( (\Delta_l, \Delta_0) \) is determined as the minimum distance over all match candidates.
\[ E(\Delta_i \Delta_j) = \sum_{i} \sum_{j} (g_i(i + \Delta_i, j + \Delta_j) - r_{k-1}(i, j))^2 \]
\[ (\Delta_l, \Delta_0) = \arg \min_{\Delta_I \Delta_j} E(\Delta_i \Delta_j) \]

While an exhaustive search through all possible template matches would be computationally expensive, the proposal is more efficient since \( \Delta_i \) and \( \Delta_j \) are constrained to a limited range on the basis of the tracking prediction. If we denote \( (\Delta_i', \Delta_j') \) as the predicted offset of the object being occluded, then \( \Delta_l \) and \( \Delta_j \) are required changing within \( [\Delta_l/2, 3\Delta_l/2] \) and \( [\Delta_j/2, 3\Delta_j/2] \), respectively, i.e. a small search window. Fig. 4 shows results of foreground detection and motion correction in four continuous frames when a flying ball is merged with players. The original method [31] of partial observations correctly follows player IDs #7 and #8 through their mutual occlusion. However, the trajectory of the ball with object ID #10 is wrongly estimated and becomes discontinuous. When the correction process is applied, the whole trajectory of the ball is accurately estimated. In addition, some new techniques like Bayesian network inference and multiple hypothesis tracking may be utilized for further robustness [34,35].

4. Identification of the ball trajectory

The method described in Section 3 will generate tracks from each camera view corresponding to the movement the ball, players (and player fragments), and other clutter such as windblown litter. In this Section, techniques are described that use visual features and motion information to estimate a measure of relative likelihood that any given track represents the motion of the ball. The domain knowledge introduced in Section 2 provides spatial–temporal constraints that can be used to track forward and back through the sequence of trajectories to maintain the identification through cluttered sequences of play.

4.1. Forward filtering

After the tracking process, in each frame every tracked object \( \mathbf{O} \) is assigned with a tracked state, including position, size, age and an estimated velocity in the image-plane. Measurements of size and velocity are further expressed in ‘ground plane’ co-ordinates, using
the homography provided by calibrated cameras and assuming that all objects lie on the ground plane.

In the proposed identification process, size, color, velocity and longevity features are used to discriminate the ball from other objects in a two part estimate $D(\mathbf{o},k)$ of the likelihood that, at frame $k$, object $\mathbf{o}$ represents the ball. The first part, $D_1(\mathbf{o})$, only uses size and color features of the object; while the second part, $D_2(\mathbf{o})$ uses motion features. The final estimate $D(\mathbf{o})$ is a weighted combination of the results from $D_1(\mathbf{o})$ and $D_2(\mathbf{o})$.

$$D(\mathbf{o},k) = \eta D_1(\mathbf{o},k) + (1-\eta) D_2(\mathbf{o},k)$$

where $D_1(\mathbf{o}), D_2(\mathbf{o}) \in [0,1]$, and $\eta \in [0,1]$ is a weight and we simply used $\eta = 1/2$.

For two reasons, the size of the moving ball may be over-estimated. Firstly, there is a motion blur effect caused by finite shutter speed. Second, if the moving ball is above the ground plane then the transformation to world co-ordinates will over-estimate the size since it assumes the ball lies on the ground plane. To accommodate this first effect, the expected size of the ball can be adapted to be a function of the velocity. Writing the image-plane velocity at frame $k$ as $(v_x(k), v_y(k))$, the expected width $w$ and height $h$ in frame $k$ are

$$w(k) = d_0 + v_x(k) \Delta T$$
$$h(k) = d_0 + v_y(k) \Delta T$$

where $d_0 = 0.22$ m is defined in Section 2.2 as the diameter of a stationary ball, and $\Delta T$ is the temporal aperture. This expression is re-arranged to define the corrected measurements $\hat{w}(k)$ and $\hat{h}(k)$ as $w(k) - v_x(k) \Delta T$ and $h(k) - v_y(k) \Delta T$, respectively. Then, $D_1$ is defined as

$$D_1(\mathbf{o},k) = \begin{cases} 0 & \text{if } \hat{w}(k) \geq 2d_0 \lor \hat{h}(k) \geq 2d_0 \lor \eta_t < 0.3 \\ \left(1 - \frac{\hat{w}(k)-d_0}{d_0}\right)\left(1 - \frac{\hat{h}(k)-d_0}{d_0}\right) & \text{otherwise} \end{cases}$$

where $\eta_t$ refers to a percentage of white pixels in the bounding box. Eq. (10) is presented as a hypothetical distribution of the likelihood that track $\mathbf{o}$ represents the ball, given its width and height, and color or properties. Alternatively, the parameters for this or another form of distribution could be estimated from the data, given sufficient training samples.

The second part of the overall expression uses the object’s absolute velocity $|v|$ and longevity $\eta_t$ as below to incorporate the observation that false alarms tend to be short-lived, and the ball tends to be rapidly moving. For this estimate of likelihood, the dependency on longevity $\eta_t$ is approximated by an exponential distribution:

$$D_2(\mathbf{o}) = \frac{|v|}{v_{\text{max}}} \left(1 - e^{-\eta_t T_0}\right)$$

where $v_{\text{max}}$ is the maximum absolute velocity of all the objects at frame $k$ (including the ball and non-ball objects), and $T_0$ is a constant. Typically the moving ball moves more quickly than players and is often the fastest moving object.

Fig. 5 shows all observed tracks, using the likelihood ball measure to color each trajectory, in about $30$ s of video sequence from camera #4. In Fig. 5, time $t$ (or frame number $k$) moves from left to right, and the horizontal image co-ordinates of the object centroids, $c_0$, is plotted up the $y$-axis, while the vertical image co-ordinate is omitted from this diagram. Trajectories in red, green and light gray show tracks that are highly likely (>0.75), possible (between 0.35 and 0.75), and unlikely (<0.35) to be the ball, respectively. From Fig. 5 we can see that the filtered results of the ball suffer discontinuous trajectories and quite a few false alarms, and the latter can be found as multiple possible ball candidates at a certain frame. These problems are mainly caused by occlusions and false alarms, and solutions to solve these two drawbacks are discussed below.

4.2. Occlusion-reasoning and tracking-back

As discussed in Section 2, there are false alarms caused by, for example, white field lines and body parts of players. When the
frequently occluded ball cannot be tracked successfully, its trajectory becomes discontinuous and even incorrect due to these false alarms. In such a context, tracking-back is a complementary process to cope with ambiguity introduced in forward filtering, especially for the cases when the ball is occluded and then comes out again. In this process, the Kalman state information such as ‘possessed’ and ‘moving-free’ can be used to infer the likely path through periods of extended occlusions.

Firstly, occlusion-reasoning and tracking-back is employed to determine whether the ball is still moving or being possessed when it is merged with other objects like player(s) during tracking. Consider a tracked ball $B_k$, which at frame $k$ is started to be occluded by a player $P_j$. At this instant of time, there is no distinct observation for $B_k$, and it is unknown whether it has entered into a possessed state, or else there was no interception, and the ball will continue the trajectory. Thus the state of $B_k$ at frame $k$ can be either moving-free or possessed. If, after $k_0$ frames or less, a new ball candidate is detected near $P_j$, then we infer that the ball has never changed from its original (moving-free) state and thus its trajectory is estimated by linear interpolation between the preceding and the following frames. When $k_0$ frames or more, a new ball candidate is detected near $P_j$, then we infer that the ball has been possessed at least $k_0$ frames prior to. When $k_0$ frames after $P_j$, then we infer that the ball has been possessed by a player $P_j$.

Secondly, tracking-back is applied to each new ball candidate $B_i$, that is instantiated inside a distance $\theta_{th}$ of the image boundary, i.e. did not appear at the edge of the image. The value of $\theta_{th}$ is set to be the maximum expected displacement (between frames) of the freely moving ball: a value of 15 pixels was used. It is assumed there is at least one player $P_j$ that is responsible for the change of state from possessed to ‘moving-free’—and the absence of ball observations when in the former state. Thus a new ball should always emerge from a player who possessed it; otherwise it is considered to be a false alarm. This $P_j$ is simply decided as a player who is closest to $B_i$, and the path of $P_j$ is then utilized as an estimate of the trajectory of $B_i$.

Fig. 5. Examples of about 30 s of tracking data and filtered ball from camera #4, in which time $t$ moves from left to right, and the horizontal image co-ordinates of the object centroids, $c_m$ is plotted up the y-axis. Red, green and light gray trajectories refer to highly likely ball, possible ball and non-ball objects. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this paper.)

5. Determining 3D ball trajectory

To estimate the 3D ball trajectory, fusion of tracked results from multiple cameras is required. Techniques for tracking players from multi-cameras are discussed in [30]. Below, a 3D trajectory model is described. This is used to provide estimates of the 3D position of the ball.

5.1. Determining 3D ball positions

If a point in a 3D co-ordinate system $b$, lying somewhere above the ground plane, is observed from two cameras $c_1$ and $c_2$ with projected positions $b_1$ and $b_2$ on the ground plane, then we can estimate $b$ from $b_1$, $b_2$, $c_1$ and $c_2$. Let $l_1$ and $l_2$ be two lines from $c_1$ to $b_1$ and $c_2$ to $b_2$, respectively. In the noise-free case they will intersect at a point $p$, and this point can be recovered geometrically. However, $l_1$ and $l_2$ usually have no intersection point due to errors caused by camera calibration and various sources of measurement noise. We can instead consider the shortest line joining $l_1$ and $l_2$—it is reasonable to place the estimate of $b$ somewhere on this line. The simplest strategy, given below, is to set the estimate as the mid-point of this line, although a different strategy may be optimal if noise properties or relative distances are taken into account.

Two points $p_1$ and $p_2$ are defined on lines $l_1$ and $l_2$, respectively, and we require the line from $p_1$ to $p_2$ as a common perpendicular of $l_1$ and $l_2$. Then $b$ should be on the line between $p_1$ and $p_2$. If $p_1$, $p_2$, $c_1$ and $c_2$ are the co-ordinate vector of the points $p_1$, $p_2$, $c_1$ and $c_2$, then $b$ can be determined by [33]:

$$\begin{align}
(b_m - c_m) \times (c_m - p_m) &= 0 \\
(b_m - c_m) \cdot (p_m - p_n) &= 0
\end{align}$$

where $m$ ranges over the two line indices $\{1,2\}$. Eq. (12) constrains $b_m$, $p_m$ and $c_m$ to all lie on the same line of $l_m$, and Eq. (13) defines the line between $p_1$ and $p_2$ to be perpendicular to $l_1$ and $l_2$.

Assuming the two cameras have same measurement covariance, then 3D ball position $b$ is simply estimated as the middle point of $p_1$ and $p_2$. Moreover, if the ball is observed in more than two cameras, we will first find the estimated 3D ball position of each pair of different views, and the final ball position $b$ is estimated as the average of these estimated points. Strategies on how to locate 3D ball positions from single view can also be found in [32].
5.2. Estimating 3D ball trajectory

With only two estimated 3D ball positions, \( r \) and \( s \), the 3D ball trajectory is obtained as follows. Let \((x(t), y(t), z(t))\) denote the 3D position of the ball at time \( t \), and \((x_r, y_r, z_r)\) and \((x_s, y_s, z_s)\) denote 3D co-ordinates of \( r \) and \( s \). We also denote \( t_r \) and \( t_s \) as the corresponding time moments, respectively. Disregarding the friction caused by the air and the ground, we can reasonably assume that the horizontal and vertical velocities are constant during the period from \( t_r \) to \( t_s \). Then, we simply have

\[
x(t) = x_r + \frac{x_s - x_r}{t_s - t_r} (t - t_r)
\]
\[
y(t) = y_r + \frac{y_s - y_r}{t_s - t_r} (t - t_r)
\]

If the ball is moving on the ground, i.e. either rolling or being possessed, we have \( z(t) = 0 \). Otherwise, a flying ball will generate a parabola trajectory satisfying Eq. (15), where \( g \) is the gravity acceleration.

\[
z(t) = -\frac{g}{2} (t - t_1)(t - t_2) + \left( \frac{z_s - z_r}{t_s - t_r} \right) t + \left( \frac{z_s t_s - z_r t_r}{t_s - t_r} \right)
\]

It is worth noting that in both Eqs. (14) and (15), the 3D ball trajectory is simply determined without any velocity information. If more than two ball positions have been located, the trajectory parameters are over-determined and a least-squares estimate can be used.

5.3. Feedback for 2D ball detection and tracking

With estimated 3D ball positions, the 2D ball detection process can be further improved. Firstly, these 3D ground co-ordinates of the ball are mapped to the image-plane and provide an estimate of new positions of the ball, again using the calibrated camera model. Then, several optimal camera views are selected in which the ball is found close enough to the corresponding camera. Meanwhile, the ball is only detected from these optimal views within a small range around the estimated new position. However, detection of the ball is resumed in all camera views once a valid ball cannot be found in selected optimal views.

6. Results and discussion

The proposed model has been tested in an 8-camera system on long sequences captured from real soccer games. In each sequence, we have 4800 frames (192 s) in miniDV format (25 frames per second, 720 x 576 resolution, 4:2:0 color depth with DCT compression for 25 Mb stream). To quantitatively evaluate the proposed method, two sets of manual ground truth data are defined. The first includes image-plane bounding box of the ball and its centroid, and the second is ground plane ball positions. In total there are 220 ground plane positions and 826 bounding boxes defined. Ball positions in other frames are then linearly interpolated by using this manual ground truths. Quantitative evaluation on 2D and 3D trajectory analysis is presented below.

6.1. Evaluating 2D performance

Firstly, 2D performance is evaluated by a measure of recovery rate \( R \), which is defined in each frame by comparing the bounding box of detected (tracked) ball \( b_{nk} \) with that of the ground truth \( g_{nk} \). Then, the common area of these two bounding boxes is extracted and divided by the area of detected blob. Let \( Area(b_{nk}) \) specify the corresponding bounding box, and this recovery rate \( R \) is then defined as

\[
R(b_{nk}) = \frac{Area(b_{nk}) \cap Area(g_{nk})}{Area(b_{nk})}
\]

To obtain an overall figure \( R(b_{nk}) \) is averaged over frames of all ground truth objects, and expressed as a percentage. When there is no buffering and tracking-back, the overall recovery rate is only 57.6%. All ball observations are either isolated from or merged with the players or referees. The corresponding recovery rates are 78.5% and 24.4%. When tracking-back is introduced, this overall recovery rate becomes 78.4%, 81.1%, 81.4% and 81.6% when the buffer size \( k_0 \) is set as 25, 50, 75 and 100 frames, respectively. The improved
tracking performance is largely achieved through more successful tracking of the ball when its appearance is merged with other objects. For example, using a buffer size of 50 frames, the detection rate in these cases is 68.5%, compared with 24.4%, in the case of no buffer. These experiments suggest that the ‘tracking-back’ technique is useful in recovering ball samples when they appear merged with players, and for isolated ball samples, its contribution is very limited. On balance, it seems that 50 frames of buffering is a good trade-off between correct tracking rates and the need for short latencies in a live stream.

Fig. 7 illustrates ball detection and tracking results in the sequence #1 from frame #897 to frame #1011, and only part of the frame images are displayed. Initially, the ball (ID #1) is detected in (a) as a new appearing object with likelihood 0.75 when it is kicked off by the player (ID #9). Then, it is in normal moving-free state with likelihood 0.90 in (b). In (c), the ball moves out of this camera view. It returns to the view in (d) and is then possessed by a new player #14, though it cannot be detected until it leaves the player in (e). In (e) and (f), the ball of ID #16 is identified again as new appearing and normal moving-free, with likelihood of 0.75 and 0.95, respectively. In (g), the ball of likelihood 0.99 is merged with player #9 and possessed. Next, a new ball of ID #11 is detected in (h) with likelihood 0.60, and finally in (j) it is found in normal moving-free status of likelihood 0.85 before it flies out of view again. These tracking IDs are dynamically assigned to each new object which has appeared in the whole frame, and these results are obtained without tracking-back. From Fig. 7 we can see that the proposed approach is very effective in detecting the ball in cases where a conventional approach is not so successful, e.g. if it is merged with players.

6.2. Evaluating 3D performance

For 3D ball tracking from multiple cameras, the distance (error) between the estimated and ground truth ball positions in ground plane is measured. We denote $N_d(x)$ as the number of detected balls within the distance of $x$ to the ground truth, and express the maximum distance allowed as $x_{mde}$. Therefore, $N_d(x_{mde})$ means the total number of valid 3D ball positions recovered. Meanwhile, denote the maximum calibration error between the eight cameras in the system as $x_{nice}$. In the experimental dataset, this is estimated to be 2.5 m. The maximum recorded error $x_{ncre}$ is $2x_{nice}$.

Let $N_0$ be the total number of frames with common timestamp tracked in the eight sequences, except those in which the ball is out of play, i.e. number of frames with an in-play ball. An overall recovery rate $V_d$ can be defined as follows.

$$V_d = N_d(x_{ncre})/N_0$$

To measure the accuracy in 3D tracking, we define an accuracy rate $A_d$ as follows.

$$A_d(x) = N_d(x)/N_d(x_{mde})$$

which refers to a percentage of recovered ball positions which have errors less than $x$ in the total number of valid 3D ball positions. Also, we take $A_d(x_{nice})$ as a reasonable measurement of 3D tracking accuracy, as it corresponds to a percentage of ball positions which have an error less than $x_{nice}$.

When there is no buffering in ball filtering, we have $V_d = 33.8\%$ and $A_d(x_{nice}) = 94.1\%$, which means very limited ball positions are recovered in high accuracy. When a buffer of 25 frames is used, we have $V_d = 72.0\%$ and $A_d(x_{nice}) = 91\%$. When the buffer size increases to 50 and 75 frames, respectively, the corresponding recovery rate and accurate rate are $V_d = 84.2\%$ and $A_d(x_{nice}) = 89.2\%$, and $V_d = 84.6\%$ and $A_d(x_{nice}) = 88.5\%$. Also, it suggests that 50 frames of latency is a good trade-off between an overall recovery rate and a high accuracy.

Fig. 8 illustrates tracking results from multiple cameras at frame #820, in which both the trajectories of the ball and players are shown. A ground plane visualization of the game is plotted in the middle and surrounded with results from the eight separate camera views. The projected 3D ball trajectory is represented in magenta, whilst the associated 2D trajectories are in gray. The ball trajectory filtered from single cameras can be found from views of cameras #3, #2 and #6.

7. Conclusions

We have proposed a novel method for soccer ball detection and tracking from real video sequences. The domain knowledge is an important component in the process model. A local matching pro-
cess is proved to be effective in compensating the Kalman tracker to deal with merged balls. Motion information and modeling the expected appearance of a moving ball have significantly improved the detection accuracy. Moreover, the application of occlusion-reasoning and tracking-back results in significant improvements of the tracking accuracy and continuity of the ball trajectory. The effectiveness of the tracking-back approach is dependent on the size of the buffering. By comparing results for different buffer sizes an appropriate trade-off between the accuracy and latency is also suggested. In our 3D trajectory model, the ball motion can be estimated with only two 3D ball positions without any velocity information. Future work includes the investigation of more complex modeling of game events for content-based understanding of soccer.

Acknowledgment

This work formed part of the INMOVE project, supported by the European Commission IST 2001-37422.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at doi:10.1016/j.cviu.2008.01.007.

References