# Tracking the Soccer Ball using Multiple Fixed Cameras

Jinchang Ren[1,*], James Orwell[2], Graeme A. Jones[2], Ming Xu[3]

1 Centre for Vision, Speech and Signal Processing, University of Surrey, U.K.
j.ren@surrey.ac.uk npurjc@yahoo.com

2 Digital Imaging Research Centre, Kingston University, U. K.
{j.orwell, g.jones}@kingston.ac.uk

3 Signal Processing Lab, Department of Engineering, Cambridge University, U. K.
mx204@eng.cam.ac.uk

**Abstract.** This paper demonstrates innovative techniques for estimating the trajectory of a soccer ball from multiple fixed cameras. Since the ball is nearly always moving and frequently occluded, its size and shape appearance varies over time and between cameras. Knowledge about the soccer domain is utilised and expressed in terms of field, object and motion models to distinguish the ball from other movements in the tracking and matching processes. Using ground plane velocity, longevity, normalized size and colour features, each of the tracks obtained from a Kalman filter is assigned with a likelihood measure that represents the ball. This measure is further refined by reasoning through occlusions and back-tracking in the track history. This can be demonstrated to improve the accuracy and continuity of the results. Finally, a simple 3D trajectory model is presented, and the estimated 3D ball positions are fed back to constrain the 2D processing for more efficient and robust detection and tracking. Experimental results with quantitative evaluations from several long sequences are reported.

**Index Terms:** Motion analysis; domain knowledge modelling; trajectory modelling; 3D vision; video signal processing, sports analysis.

# 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 colour 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 colour when moving at speed (see examples in Figure 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.

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 centre. However, due to complex camera movements and partial views of the field, it is hard to obtain accurate camera parameters for on-field ball positioning. In Gong *et al* [1], white colour and circular shape are employed to detect balls in image sequences. In Yu *et al* [15], candidate balls are first identified by size range, colour 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 filter are used to track balls after manual initialization. In Liang *et al* [17], colour, 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 colour and size features, however, it fails in dealing with cases of occlusion or small size of the ball. Since colour and shape varies considerably in soccer games (see Fig. 1), these methods seem unlikely to provide robust solutions.

Using multiple fixed cameras has the advantage that calibration is easier to establish and that accurate on-field positions can be extracted 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 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 white colour like a ball.

As 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 is utilized. In Ohno *et al* [16], 3D ball trajectory is modelled by considering air friction and gravity but depends on 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. These are unlikely to be robust as the shadow positions depend more on light positions than camera projections.

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 modelled 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 modelled 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 taken as feedback in 2D processing 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 modelled. 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, 3D ball trajectory model is discussed along with approaches on 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 Modelling of Soccer Games

Domain knowledge including colour, ball shape and pitch geometry is widely adopted in soccer and other sports video analysis systems [1-3, 12, 14-16, 22]. In 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 modelling such scenarios for context-based vision [26], or closed world tracking [6]. According to Strat and Fischler [26] and Initille 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 modelled in several aspects and discussed as follows.

## 2.1 Field Modelling

In real soccer games, the play field (or 'pitch') can be exactly modelled with many corner points of given locations. This pitch model is useful in camera calibration to obtain bi-directional co-ordinate 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 modelled as a mask image to remove potential false alarms caused by these lines. The FOVs of all eight cameras are shown in Figure 2(a). Figure 2(b) and 2(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 Modelling

The objects in soccer context include a single ball and 25 persons comprising a referee and his two assistants, two goalkeepers and 20 players from two teams, namely Teams A and B. For a multi-camera system, observations of these people can be categorized into these five classes according to colours of their uniforms. The ball is the sixth category, and a seventh category is defined as 'anything else' (windblown litter, flying turf, pitch markings judged as motion from camera-shake, people warming up on touchline, etc.).

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 colour depends on its condition and requirements from competitions; for the matches recorded for these experiments (English Premiership) its initial colour 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 will have small or big error for objects on or above the ground plane, unless its height is known from some other source.

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.

Figure 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 Modelling

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. In fact, 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 the ball 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 re-placement balls to accelerate the process of re-introducing the ball into play. This is an allowable dis-continuity in the trajectory of 'the ball' (though 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], $\left(\boldsymbol{\mu}_k^{(l)}, \sigma_k^{(l)}, \omega_k^{(l)}\right)$, is used to estimate an initial background, where $\boldsymbol{\mu}_k^{(l)}$, $\sigma_k^{(l)}$ 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 second stage, this initial background image is continuously updated using a faster running average algorithm for efficiency [30]:

$$\boldsymbol{\mu}_k = [\alpha_L \mathbf{I}_k + (1-\alpha_L)\boldsymbol{\mu}_{k-1}]F_k + [\alpha_H \mathbf{I}_k + (1-\alpha_H)\boldsymbol{\mu}_{k-1}]\overline{F_k} \tag{1}$$

where $0 < \alpha_L << \alpha_H << 1$, and $F_k$ 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 $\| \mathbf{I}_k - \boldsymbol{\mu}_{k-1} \|$ 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, which prevents estimation errors accumulating in a hierarchy of Kalman filters. Besides, all detected small objects (less than $0.1\,m^2$) 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:

$$\mathbf{x}_I = [r_0 \quad c_0 \quad \dot{r}_0 \quad \dot{c}_0 \quad \Delta r_1 \quad \Delta c_1 \quad \Delta r_2 \quad \Delta c_2]^{\mathrm{T}} \tag{2}$$

$$\mathbf{z}_I = [r_0 \quad c_0 \quad r_1 \quad c_1 \quad r_2 \quad c_2]^{\mathrm{T}} \tag{3}$$

where $(r_0, c_0)$ is the centroid, $(\dot{r}_0, \dot{c}_0)$ is the velocity, $(r_1, c_1)$ and $(r_2, c_2)$ are the top-left and bottom-right corners of the bounding box, respectively ($r_1 < r_2$ and $c_1 < c_2$); $(\Delta r_1, \Delta c_1)$ and $(\Delta r_2, \Delta c_2)$ are the relative positions of $(r_0, c_0)$ to $(r_1, c_1)$ and $(r_2, c_2)$.

The state transition and measurement equations in the Kalman filter are:

$$\begin{aligned} \mathbf{x}_I(k+1) &= \mathbf{A}_I \mathbf{x}_I(k) + \mathbf{w}_I(k) \\ \mathbf{z}_I(k) &= \mathbf{H}_I \mathbf{x}_I(k) + \mathbf{v}_I(k) \end{aligned} \tag{4}$$

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). Besides, $\mathbf{I}_2$ and $\mathbf{O}_2$ represent $2 \times 2$ identity and zero metrics.

$$\mathbf{A}_I = \begin{bmatrix} \mathbf{I}_2 & \Delta T \cdot \mathbf{I}_2 & \mathbf{O}_2 & \mathbf{O}_2 \\ \mathbf{O}_2 & \mathbf{I}_2 & \mathbf{O}_2 & \mathbf{O}_2 \\ \mathbf{O}_2 & \mathbf{O}_2 & \mathbf{I}_2 & \mathbf{O}_2 \\ \mathbf{O}_2 & \mathbf{O}_2 & \mathbf{O}_2 & \mathbf{I}_2 \end{bmatrix} \qquad \mathbf{H}_I = \begin{bmatrix} \mathbf{I}_2 & \mathbf{O}_2 & \mathbf{O}_2 & \mathbf{O}_2 \\ \mathbf{I}_2 & \mathbf{O}_2 & \mathbf{I}_2 & \mathbf{O}_2 \\ \mathbf{I}_2 & \mathbf{O}_2 & \mathbf{O}_2 & \mathbf{I}_2 \end{bmatrix} \tag{5}$$

The Mahalanobis distance is used to associate each observation to (at most) one tracked object, and the distance between the $i^{\text{th}}$ observation $\mathbf{O}_i$ and the $j^{\text{th}}$ tracked object $\mathbf{O}'_j$ is given in Eqn 6, where $\mathbf{\Sigma}$ is the covariance matrix.

$$\delta_{ij}^2 = (\mathbf{O}_i - \mathbf{O}'_j)^{\mathrm{T}} \mathbf{\Sigma}^{-1} (\mathbf{O}_i - \mathbf{O}'_j) \tag{6}$$

During the tracking process, several states are defined to identify different cases. These states are: *new*, *normal*, *merged*, *missing* and *terminated*, and are further explained as follows. For each existing tracked object, if it has corresponding observation matched, we set it in *normal* state. Otherwise, it is marked as *missing* and updated by predicted state estimation. If an object has been *missing* for more than $M$ frames, it is *terminated*. All unmatched observations are identified as *new* objects in tracking. If different objects share common regions in their bounding boxes, they are *merged*. Moreover, the *age* of a track is number of the frames that an object has been tracked, and the *age* of a *new* object is 1.

### 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 $0.3\,m^2$) is found being merged with another (larger) object of area more than $0.4\,m^2$. 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 $r_{k-1}$ and $g_k$ 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 $(\Delta i_0,\Delta j_0)$ is determined as the minimum distance over all match candidates.

$$\begin{aligned}&E^2(\Delta i,\Delta j)=\sum_i\sum_j(g_k(i+\Delta i,j+\Delta j)-r_{k-1}(i,j))^2\\&(\Delta i_0,\Delta j_0)=\arg\min_{\Delta i,\Delta j}(E(\Delta i,\Delta j))\end{aligned} \tag{7}$$

Though traditional template matching is a (computationally expensive) exhaustive search, the proposal is more efficient because $\Delta i$ 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 $\Delta i$ and $\Delta j$ are required changing within $\Delta i'\pm\Delta i'/2$ and $\Delta j'\pm\Delta j'/2$, respectively. Figure 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 the previous Section 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 $\mathbf{o}_i$ 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*, *colour*, *velocity* and *longevity* features are used to discriminate the ball from other objects in a two-part estimate $D(\mathbf{o}_i, k)$ of the likelihood that, at frame *k*, object $\mathbf{o}_i$ represents the ball. The first part, $D_1()$, only uses size and colour features of the object; while the second part, $D_2()$ uses motion features. The final estimate $D()$ is a weighted combination of the results from $D_1()$ and $D_2()$.

$$D(\mathbf{o}_i, k) = \eta D_1(\mathbf{o}_i, k) + (1-\eta) D_2(\mathbf{o}_i, k) \tag{8}$$

where $D_1(), D_2() \in [0,1]$, 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-coordinates will over-estimate the size since it assumes the ball lies on the ground plane. To accommodate the 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

$$\begin{aligned} w(k) &= d_0 + v_x(k)\Delta T' \\ h(k) &= d_0 + v_y(k)\Delta T' \end{aligned} \tag{9}$$

where $d_0 = 0.22m$ 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

$$D_1(\mathbf{o}_i,k)=\begin{cases}0 & if \quad \hat{w}(k)\geq 2d_0 \vee \hat{h}(k)\geq 2d_0 \vee \eta_c < 0.3\\ \left(1-\dfrac{|\hat{w}(k)-d_0|}{d_0}\right)\left(1-\dfrac{|\hat{h}(k)-d_0|}{d_0}\right) & otherwise\end{cases} \tag{10}$$

where $\eta_c$ refers to a percentage of white pixels in the bounding box. Eqn. 10 is presented as a hypothetical distribution of the likelihood that track $\mathbf{o}_i$ represents the ball, given its width and height, and colour 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 $|\mathbf{v}_i|$ and longevity $n_i$ 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 $n_i$ is approximated by an exponential distribution:

$$D_2(\mathbf{o}_i)=\frac{|\mathbf{v}_i|}{v_{\max}}(1-e^{-n_iT_0}) \tag{11}$$

where $v_{\max}$ 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.

According to the tracked evidences, Figure 5 shows filtered results of the ball trajectory in about 30 seconds 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 grey are from highly likely ball (with a likelihood greater than 0.75), possible ball (with a likelihood between 0.35 and 0.75), and non-ball objects (players or false alarms), 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 like field line noise and body parts of players which may appear like a ball. 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 $B_i$, 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 $B_i$ 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 $B_i$ 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 two positions when it is just before and after occlusion with $P_j$. Otherwise, it should be defined as possessed by $P_j$ from frame *k* and afterwards and the path of $P_j$ is then utilized as an estimate of the trajectory of $B_i$.

Secondly, tracking-back is applied when each *new* ball candidate $B_i$ is found not instantiated on the image boundary. 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 taken as estimated trajectory of $B_i$ during occlusion (in possession). Moreover, if a candidate ball has longevity less than a given threshold, for example 4 frames, it is also considered as a short-lived false alarm (caused by imperfect foreground detection, as discussed in Section 2.2).

Apparently, the above tracking-back procedure requires at least $k_0$ frames of ‘hindsight’ to classify the state based on subsequent observations. Therefore, a buffer of more than $k_0$ frames is introduced to store the tracking results and states before making a final estimate of the play state and ball trajectory. It is interesting to note that the ball trajectory obtained here is 2D results which corresponding to projected 3D trajectory on the ground plane, and how to use these 2D results to determine 3D trajectory of the ball is discussed in Section 5.

Figure 6 plots improved ball trajectory after occlusion reasoning and tracking-back, in which the bottom image is the final result after tracking-back; hence there is a considerable improvement in the accuracy of detection and continuity of trajectories. Comparing the final ball trajectory in Fig 6 with Fig 5 clearly demonstrates that this post-processing has effectively recovered the ball positions even when it is occluded (being possessed with players). At the same time, false alarms of short lived objects are dramatically reduced.

## 5 Determining 3D Ball Trajectory

To estimate 3D ball trajectory, fusion of tracked results from multiple cameras are required. Techniques for tracking players from multi-cameras are discussed in [30], and as for the ball, a 3D trajectory model with details on how to decide 3D ball positions is presented below.

### 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 $b$ 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 $p_2$ be a common perpendicular of $l_1$ and $l_2$. Then $b$ should be on the line between $p_1$ and $p_2$. If we denote $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{c}_1$ and $\mathbf{c}_2$ as the co-ordinate vector of the points $p_1$, $p_2$, $c_1$ and $c_2$, then $\mathbf{p}_1$ and $\mathbf{p}_2$ can be determined by [33]:

$$(\mathbf{b}_m - \mathbf{c}_m) \times (\mathbf{c}_m - \mathbf{p}_m) = 0 \tag{12}$$

$$(\mathbf{b}_m - \mathbf{c}_m) \cdot (\mathbf{p}_m - \mathbf{p}_m) = 0 \tag{13}$$

where $m$ ranges over the two line indices $\{1,2\}$. Equation (12) constraints $b_m$, $p_m$ and $c_m$ to all lie on the same line of $l_m$, and equation (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 [33].

## 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 air friction, 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

$$
\begin{aligned}
x(t) &= x_r + \frac{x_s - x_r}{t_s - t_r}(t - t_r) \\
y(t) &= y_r + \frac{y_s - y_r}{t_s - t_r}(t - t_r)
\end{aligned}
\tag{14}
$$

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 Eqn. 15, where $g$ is the gravity acceleration.

$$
z(t) = -\frac{g}{2}(t - t_r)(t - t_s) + \left(\frac{z_s - z_r}{t_s - t_r}\right)t + \left(\frac{z_r t_s - z_s t_r}{t_s - t_r}\right) \tag{15}
$$

It is worth noting that in both Eqns. 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 coordinates of the ball are mapped to image plane and provide an estimate of new positions of the ball, again using our calibrated camera model. Then, several optimal camera views are selected in which the ball is predicted of high visibility, and this is determined if 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.

Moreover, an adaptive frame dropping is also applied for efficiency. For these optimal views with the ball contained, detection and tracking is completed frame by frame, i.e. without frame dropping. Otherwise, foreground detection and tracking is only employed in every $\eta_d$ frames. The reason is that players may move fast or slowly when they are near or away from the ball, hence the adaptive frame dropping scheme. If the 3D ball position is invalid, then the ball is detected from all camera views without frame dropping.

## 6 Results and Discussions

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 seconds) in miniDV format (25 frames per second, $720 \times 576$ resolution, 4:2:0 colour depth with DCT compression for 25 Mb stream). To quantitatively evaluate the proposed method, two groups 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 detection rate $R$, which is defined in each frame by comparing the bounding box of detected (tracked) ball $b_{n,k}$ with that of the ground truth $g_{n,k}$ from sequence $n$ at frame $\# k$. Then, the common area of these two bounding boxes is extracted and divided by the area of detected blob. Let $Area(\cdot)$ specify the corresponding bounding box, and this detected rate $R$ is then defined as

$$R(b_{n,k}) = \frac{Area(b_{n,k}) \bigcap Area(g_{n,k})}{Area(b_{n,k})} \tag{16}$$

To obtain an overall figure $R(b_{n,k})$ is averaged over frames of all ground truth objects. When there is no buffering and tracking-back, the overall detection rate is only 57.6%. As for ball observations that are isolated from or merged with the players, namely isolated or merged balls, the corresponding detection rates are 78.5% and 24.4%. When tracking-back is introduced, this overall recover rate becomes

78.4%, 81.1%, 81.4% and 81.6% when the buffer size is set as 25, 50, 75 and 100 frames, respectively. As for isolated balls, their average detection rates are 84.1%, 84.9%, 85.1% and 85.2%. While for merged ball samples, the corresponding detection rates are 63.7%, 68.5%, 68.7% and 69.0%. Furthermore, more than 90% of correctly detected balls are found within a six pixels deviation between the centroid points of the two bounding boxes. 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.

Figure 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 $x_{mce}$. In the experimental dataset, this is estimated to be 2.5 meters. The maximum recorded error $x_{mde} = 2x_{mce}$.

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. An overall recovery rate $V_d$ can be defined as follows.

$$V_d = N_d(x_{mde}) / N_0 \tag{17}$$

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

$$A_d(x) = N_d(x) / N_d(x_{mde}) \tag{18}$$

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 $A_d(x_{mce})$ as a reasonable measurement of 3D tracking accuracy, as it corresponds to a percentage of ball positions which have an error less than $x_{mce}$.

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

Figure 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 visualisation 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 grey. The ball trajectory filtered from single cameras can be found from views of cameras #3, #2 and #6.

### 6.3 Evaluating Performance on Adaptive Frame Dropping

According to our adaptive frame dropping strategy, 3D ball positions are used as feedback in 2D processing for ball detection and tracking. As for the eight sequences in total, up to three optimal views are determined by the estimated 3D ball positions, and frame dropping occurs in the other five or more camera views.

In our system, foreground detection, image plane tracking, ball filtering and 3D positioning occupy 95.5%, 2.5%, 1% and 1% of the entire computational load needed, respectively. In dropped frames, foreground detection is omitted, and the tracking process only uses prediction for estimation, thus the overall efficiency has been well improved. When $\eta_d$ is set as 2, 3 and 4, the computational load is reduced by 49%, 68% and 78%. At the same time, the tracking accuracy is found to degrade between 1.2% and 2.6%. Therefore a lower frame rate of around 6 frames per second can still provide accurate tracking performance.

## 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 process is proved effective in compensating the Kalman tracker to deal with merged balls. Motion information and modelling 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. Finally, feedbacks from 3D ball positioning to 2D detection and tracking seems more efficient. Future work includes the investigation of more complex modelling of game events for content-based understanding of soccer.

## Acknowledgement

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

## References

[1] Y. Gong, T.-S. Lim, H. C. Chua, H. J. Zhang, M. Sakauchi, Automatic parsing of TV soccer programs, in: Proc. Multimedia Computing and Systems, 1995, pp. 167-174.

[2] D. Yow, B. L. Yeo, M. Yeung, B. Liu, Analysis and presentation of soccer highlights from digital video, in: Proc. ACCV, 1995, pp. 499-503.

[3] A. Ekin, M. Tekalp, R. Mehrotra, Automatic soccer video analysis and summarization, IEEE Trans. on Image Processing. 12(7) (2003) 796-807.

[4] R. Urtasun, D. Fleet, P. Fua, Monocular 3–D tracking of the golf swing, in: Proc. CVPR, vol. 1, San Diego, CA, 2005, pp. 932–939.

[5] F. Yan, W. Christmas, J. Kittler, A tennis ball tracking algorithm for automatic annotation of tennis match, in: Proc. BMVC, vol. 2, Oxford, 2005, pp. 619–628.

[6] S. Intille, J. Davis, A. Bobick, Real-time closed-world tracking, in: Proc. CVPR, San Juan, Puerto Rico, 1997, pp. 697–703.

[7] K. Okuma, A. Taleghani, N. de Freitas, J. Little, D. Lowe, A boosted particle filter: multitarget detection and tracking, in: Proc. ECCV, Prague, Czech Republic, 2004, pp. 28–39.

[8] Y. Rui, A. Gupta, A. Acero, Automatically extracting highlights for TV Baseball programs, in: Proc. ACM Multimedia, 2000, pp. 105-115.

[9] S. Nepal, U. Srinivasan, G. Reynolds, Automatic detection of 'Goal' segments in basketball videos, in: Proc. ACM Multimedia, Ottawa, 2001, pp.261-269.

[10] M. A. Zaveri, S. N. Merchant, U. B. Desai, Small and fast moving object detection and tracking in sports video sequences, in: Proc. ICME, vol. 3, 2004, pp. 1539-1542.

[11] T. Bebie, H. Bieri, SoccerMan – reconstructing soccer game from video sequence, in: Proc. ICIP, 2000, pp. 898-902.

[12] K. Matsumoto, S. Sudo, H. Saito, S. Ozawa, Optimized camera viewpoint determination system for soccer game broadcasting, in: Proc. IAPR Workshop on Machine Vision Applications, Tokyo, 2000, pp. 115-118.

[13] Y. Seo, S. Choi, H. Kim, K. S. Hong, Where are the ball and players?: soccer game analysis with color based tracking and image mosaick, in: Proc. ICIAP, 1997, pp. 196-203.

[14] T. D’Orazio, C. Guaragnella, M. Leo, A. Distante, A new algorithm for ball recognition using circle Hough transform and neural classifier, Pattern Recognition. 37(3) (2004), 393-408.

[15] X. Yu, C. Xu, H. W. Leong, Q. Tian, Q. Tang, K. W. Wan, Trajectory-based ball detection and tracking with applications to semantic analysis of broadcast soccer video, in: Proc. ACM Multimedia, 2003, pp. 11-20.

[16] Y. Ohno, J. Miura, Y. Shirai, Tracking players and estimation of the 3D position of a ball in soccer games, in: Proc. ICPR, 2000, pp. 145-148.

[17] D. Liang, Y. Liu, Q. Huang, W. Gao, A scheme for ball detection and tracking in broadcast soccer video, in: Proc. Pacific Conference on Multimedia, LNCS 3767, 2005, pp. 864-875.

[18] X. F. Tong, H. Q. Lu, Q. S. Liu, An effective and fast soccer ball detection and tracking method, in: Proc. ICPR, 2004, pp. 795-798.

[19] K. Choi, Y. Seo, Probabilistic tracking of the soccer ball, in: Proc. ECCV Workshop Statistical Methods in Video Processing, LNCS 3247, 2004, pp. 50-60.

[20] T. Kim, Y. Seo, K. S. Hong, Physics-based 3D position analysis of a soccer ball from monocular image sequences, in: Proc. ICCV, 1998, pp. 721-726.

[21] I. Reid, A. North, 3D trajectories from a single viewpoint using shadows, in Proc. BMVC, 1998, pp. 863-872.

[22] D. Zhong, S. F. Chang, Structure analysis of sports video using domain models, in: Proc. ICME, Tokyo, 2001, pp. 920-923.

[23] N. Babaguchi, Y. Kawai, T. Kitashi, Event based indexing of broadcasted sports video by intermodal collaboration, IEEE Trans. Multimedia. 4(2002) 68-75.

[24] V. Tovinkere, R. J. Qian, Detecting semantic events in soccer games: toward a complete solution, in: Proc. ICME, 2001, pp. 1040-1043.

[25] B. Li, M. I. Sezan, Event detection and summarization in American football broadcast video, in: Proc. SPIE, vol. 4676, 2002, pp. 202-213.

[26] T. M. Strat, M. A. Fischler, Context-based vision: recognizing objects using information from 2D and 3D imagery, IEEE Trans. PAMI. 13(10) (1991) 1050-1065.

[27] R. Y. Tsai, An efficient and accurate camera calibration technique for 3D machine vision, in: Proc. CVPR, 1986, pp. 364-374.

[28] J. Canny, A computational approach to edge detection, IEEE Trans. PAMI, 8(1986) 679-698.

[29] C. Stauffer, W. E. L. Grimson, Adaptive background mixture models for real-time tracking, in: Proc. CVPR, 1999, pp. 246-252.

[30] M. Xu, L. Lowey, J. Orwell, Architecture and algorithms for tracking football players with multiple cameras, IEE Proceedings - Vision, Image, and Signal Processing. 152(2005) 232–241.

[31] M. Xu, T. Ellis, Partial observation vs. blind tracking through occlusion, in: Proc. BMVC, 2002, pp. 777-786.

[32] J. Ren, J. Orwell, G.A. Jones, M. Xu, A general framework for 3D soccer ball estimation and tracking, in: Proc. ICIP, 2004, pp. 1935-1938.

[33] J. Ren, J. Orwell, G.A. Jones, M. Xu, Real-time 3D soccer ball tracking from multiple cameras, in: Proc. BMVC, 2004, pp. 829-838.

[34] P. Nillius, J. Sullivan, S. Carlsson, Multi-target tracking – linking identities using Bayesian network inference, in: Proc. CVPR, 2006, pp. 2187-2194.

[35] J. Sullivan, S. Carlsson, Tracking and labelling of interacting multiple targets, in: Proc. ECCV, 2006, pp. 619-632.

## List of Figure Captions:

**Fig. 1.** Ball samples in different size, shape and colours from same image sequences: Top and bottom rows are from two different camera views, respectively. The last sample in each row is the ball passing through field lines.

**Fig. 2**. FOVs of eight cameras in our system (a), and example images of an object-free pitch (b) with its associated field line mask (c) from camera #1.

**Fig. 3.** Enlarged images of potential objects in different colour boxes with the ball (in white), players (in blue), field line noise (in red) and body parts of players (in yellow).

**Fig. 4**. Tracking correction results in four consecutive frames (from left to right) when the moving ball of ID 10 is merged with a player (ID 8) in the sequence from camera #1: The top row is detected foreground, the middle and the bottom rows are results without and with tracking correction, respectively.

**Fig. 5**. Examples of about thirty seconds 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, $c_{0,}$ is plotted up the *y*-axis. Red, green and light grey trajectories refer to highly likely ball, possible ball and non-ball objects.

**Fig. 6.** Refined results of ball trajectory after occlusion reasoning (top) and also tracking-back (bottom). In the top image, red, green and blue trajectories refer to highly likely ball, possible ball and in-possession ball, respectively. In the bottom image, red and light grey trajectories correspond to ball and non-ball objects, and short-lived false alarms are removed.

**Fig. 7.** Ball detection (tracking) results from sequence #1 at frame #897, #904, #918, #953, #967, #980, # 1005 and #1011 (from a to j), with likelihood and tracking status shown above the corresponding bounding boxes.

**Fig. 8**: Multiview and single-view tracking results at frame #820. The surrounding images (from top-left to top-right) correspond to cameras C4, C3, C8, C2, C1, C7, C6 and C5.